On Modelling and Understanding Image Manifolds

Alan Woodland

Department of Computer Science

University of Wales

Aberystwyth

March 2010

This thesis is submitted in partial fulfilment of the requirements for the degree of

Doctor of Philosophy of The University of Wales.

Declaration

This thesis has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed ..... (candidate)

Date .....

Statement 1

This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed ..... (candidate)

Date .....

Statement 2

I hereby give consent for my thesis, if accepted, to be made available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed ..... (candidate)

Date .....

Abstract

Images of objects (e.g. a rotating teapot, views from a camera mounted on a robot, an actor illuminated from different positions on a hemisphere, etc.) which vary between them by some controlled parameter tend not to be randomly located in the space of all possible $w \times h$ pixel images. Instead, with each pixel comprising of c colour components, each image forms one $w \times h \times c$ dimensional position vector, in a space commonly referred to as image space. Instead of these images being randomly located, small changes in the sampling parameters normally produce neighbouring images in image space. This in turn can form structures, such as curves, surfaces, or more generally manifolds in image space. These are not novel, and commonly referred to as image manifolds in the literature.

To date studies of image manifolds have been limited in both scope and application. This work presents an investigation of some of the features of 28 specific collections of images from a diverse range of sources, which potentially form image manifolds. We investigate several properties of these image manifolds — their general shape, and curvature. Image manifolds are well known not to be simple linear structures, instead, as we see in our study they are highly complex shapes.

The main contribution of this work however, is a study of two novel approaches to numerical modelling of these image manifolds. Given the highly curved nature of image manifolds, which our investigation into the specific datasets confirmed, our approach is based upon extended versions of two techniques (NURBS and PDE Surfaces) typically used in the field of computer graphics to model curves and surfaces in 3-D space. Our approach is to construct exact geometric models, in image space, of the underlying image manifolds. The number of sample images required to adequately represent the appearance of an object or environment is a complex function of the subject itself. We show that it is possible to produce

acceptable (depending on the application, which we discuss briefly), compact representations of objects and environments using our models. Furthermore we show that an awareness of simple geometric properties can improve the construction of such models, by focusing on improving the model in higher curved areas.

Acknowledgements

I would like to thank all of the many people who have encouraged me and supported me during the course of my PhD. In particular I would like to thank Frédéric Labrosse, my supervisor, for reading every draft I wrote quicker than I could write them! Also my second supervisor, Mark Neal, for all his input and comments on my work. I would also like to thank the members of the VGV group, past and present, here at Aberystwyth for listening to all the talks I gave and filling me with ideas and suggestions. I would also like to thank Hassan Ugail and Gabriel Gonzales for their help during my time at Bradford, which made the PDE surface method chapter possible. The VVG network funded this work with a generous grant, without which this collaboration could not have happened.

I would like to thank my examiners Peter Hall and Peter Tiňo for the constructive criticism they provided during the viva process which helped improve and refine a great many areas of my thesis. I would also like to thank Reyer Zwiggelaar for chairing my examination and Margret Anthony for organising the viva and providing administrative support throughout the duration of my time as a PhD student in Aberystwyth.

Finally I would like to thank my friends and family for putting up with me throughout this time, in particular Val Woodland and Lisa Schaffnit for the numerous spelling, grammar and punctuation errors they helped me to find. Toby Gray also deserves a special mention for the many discussions we shared about programming and mathematics.

This work was funded by an APRS award for which I am very grateful.

Contents

1 Introduction **20**

1.1 Context 20

1.2 Contributions 22

1.3 Thesis outline 23

I Foundation **26**

2 Image Based Rendering Literature review **27**

2.1 Introduction 27

2.2 The plenoptic function 29

2.3 Explicit Geometry 31

2.3.1 MCOP 31

2.3.2 Layered Depth Images 33

2.4 Implicit Geometry 34

2.4.1 Plenoptic stitching 35

2.4.2 Image-based priors 36

2.4.3 View interpolation 36

2.5 No geometry 37

2.5.1 Lightfield and Lumigraph 37

2.5.2 Concentric Mosaics 39

2.6 Conclusion 40

2.7 Summary 43

3 Image space and image manifolds 44

3.1 Introducing image space and image manifolds 44

3.2 Image space IBR 46

3.3 Appearance based vision 49

3.3.1 Robot vision 50

3.3.2 Faces 52

3.3.3 Manifold learning and dimensionality reduction 55

3.3.4 Conclusion 62

3.4 Image manifolds studied 63

3.5 Conclusion 67

3.6 Summary 71

4 Aims and Objectives 72

4.1 Introduction 72

4.2 The research question 73

4.3 Aims and objectives 74

4.3.1 The properties of image manifolds 74

4.3.2 Representing image manifolds 75

4.4 Summary 78

5 Methodology 79

5.1 Image metrics 80

5.1.1 Fundamentals 81

5.1.2 Intrinsic metrics 82

5.1.3 Evaluating metrics 84

5.1.4 Alternative candidate metrics 85

5.1.5 Evaluation of the metrics 92

5.2 Experimental setup 97

5.3 Conclusions 100
5.4 Summary 101

II Looking at image manifolds 102

6 Measuring image manifolds 103

6.1 Midpoint distance 104
6.2 Curvature 107
6.3 Results 109
6.4 Discussion 112
6.5 Conclusions 119
6.6 Summary 121

7 Dimensionality and Visualisation 122

7.1 Introduction 122
7.2 Implementation 124
7.3 Results and Discussion 125
7.3.1 Re-projecting the manifolds 127
7.3.2 Strategies for sub-space selection 132
7.3.3 Alternative strategies 139
7.4 Summary 140

III Modelling image manifolds 141

8 Linear combinations 146

8.1 Introduction 146
8.2 Implementation 148
8.3 Experimental setup 150
8.4 Results 151
8.5 Discussion 153

8.6 Summary 162

9 NURBS **163**

9.1 Introduction 163

9.2 Traditional NURBS background 164

9.2.1 Bézier Curves 164

9.2.2 NURBS curves 167

9.2.3 NURBS surfaces 173

9.3 Beyond surfaces in 3-D space 174

9.4 Experimental setup 176

9.5 Results 177

9.6 Discussion 177

9.7 Summary 188

10 PDE method **189**

10.1 Introduction 189

10.2 Traditional PDE surfaces 190

10.3 Numerical Methods for PDE surfaces 194

10.3.1 Laplacian 194

10.3.2 Multigrid solutions 199

10.3.3 Biharmonic 200

10.4 Higher order elliptic PDEs 203

10.4.1 Analytic 204

10.4.2 Numerical 204

10.5 Higher dimension PDE surfaces 205

10.5.1 Increasing the dimensionality of the embedding space 205

10.5.2 The parameter space 206

10.6 Implementation 207

10.7 Experimental setup 210

10.8 Results 214
10.9 Discussion 216
10.9.1 Analytic 217
10.9.2 Numeric 226
10.9.3 Comparative 228
10.10Conclusion 231
10.11Summary 232

IV 'Bringing it all together' 233

11 Comparing the three models 234
11.1 Linear vs. NURBS 235
11.1.1 4-D dataset 242
11.2 Linear vs. PDE 243
11.3 NURBS vs. PDE 246
11.4 Conclusion 248
11.5 Summary 249

12 Improving the models 250
12.1 Approximation 251
12.1.1 Experimental setup 252
12.1.2 Results and Discussion 253
12.2 Midpoint distance directed construction 258
12.2.1 Experimental setup 259
12.2.2 Results and Discussion 260
12.3 Conclusions 262
12.4 Summary 263

13 Conclusion 264
13.1 Answering the question 264

13.2 General Conclusion 265
13.3 Future work 266

References **272**

Appendices **290**

A Literature summary **290**
A.1 Classic IBR 290
A.2 New IBR 292
A.3 Appearance based vision (general) 296
A.4 Faces and lips 299
A.5 Manifold learning/Dimensionality reduction 301

B Image Manifold list **306**

C Metrics for visual navigation **333**

D PCA plots of manifolds **341**

E Linear Model **380**
E.1 Summary tables 380

F NURBS model **391**
F.1 Configurations 391
F.2 Summary tables 392

G PDE **407**
G.1 Numerical Schemes for PDEs 408
G.2 Configurations 410
G.3 Summary tables 414
G.3.1 Analytic 414

G.3.2 Numeric 420

H Comparison of results **426**

H.1 PDE analytic Vs. PDE numeric 426

H.2 Linear Vs. PDE analytic 428

H.3 Linear Vs. PDE numeric 430

H.4 Linear Vs. NURBS 433

H.5 NURBS Vs. PDE 437

List of Figures

1.1 Where this thesis sits amongst related work. 22

2.1 The continuum of rendering techniques 28
2.2 Small slices of the plenoptic function 31
2.3 Example MCOP image 32
2.4 LDI example 34
2.5 Plenoptic stitching parametrisation example 35
2.6 Lightfield example 38
2.7 Concentric mosaics 39

3.1 The first four eigenfaces of a set of face images 53
3.2 Mapping a manifold of perceptual observations 57

5.1 The numberline of natural numbers. 81
5.2 Counter example (from STRAIGHT_5 dataset) for Euclidean distance as intrinsic metric for image manifolds 83
5.3 Counterintuitive result from Euclidean metric 89
5.4 Illustration of a perceptual phenomenon 91
5.5 Visual navigation metric experiment 94
5.6 Inputs for our noise test 96
5.7 Noise introduction and metrics 97
5.8 Downsampling a manifold sampled on a grid 98

6.1 Measuring deviations from straightness 106
6.2 The approximation of path curvature of Lu (1998) 108
6.3 Midpoint distance measure of the Brush image manifold 109
6.4 Midpoint distance measure of the Straight_8 image manifold . . . 110
6.5 Midpoint distance measure of the Straight_1 image manifold . . . 110
6.6 Midpoint distance measure of the Teapot image manifold 111
6.7 Curvature measure of the Brush image manifold 112
6.8 Curvature measure of the Straight_8 image manifold 113
6.9 Curvature measure of the Straight_1 image manifold 113
6.10 Curvature measure of the Teapot image manifold 113
6.11 Comparing midpoint distance and curvature 114
6.12 Visual examples of changes in curvature 118

7.1 Views of the first principal component of four of our datasets 126
7.2 The eigenvalues from four of our datasets 126
7.3 Views of the first three principal components of the BMnoise datasets 127
7.4 PCA plots of some of the datasets, using the 'impca' method 128
7.5 PCA plots of the datasets, using the 'mtfast' method 132
7.6 PCA plots of some of the datasets, 'first but one' eigenvectors . . . 134
7.7 PCA plots of some of the datasets, smallest eigenvectors 135
7.8 PCA plots of some of the datasets, using the 'mtfast' method, smallest eigenvectors 136
7.9 PCA plots of the datasets, smallest, non-zero eigenvectors. 137
7.10 PCA plots of the datasets, median ±1 eigenvectors. 138

8.1 Linear interpolation example 147
8.2 Linear interpolation example image 147
8.3 Recursive implementation of linear interpolation 150

8.4 CIRCLE_5 results . . . 154
8.5 The mean error vs gap . . . 155
8.6 An example of error not being a function of the distance of two samples . . . 157
8.7 The standard deviation of mean error, vs gap . . . 158
8.8 Hypothetical manifold, illustrating standard deviation of errors . . . 159
8.9 Detail of a stretch of WOODBOX . . . 160
8.10 Overview of WOODBOX . . . 161

9.1 Bernstein polynomials . . . 165
9.2 Bézier curve examples . . . 165
9.3 Bézier curve blending . . . 166
9.4 Knot vector influence . . . 170
9.5 Interpolation vs. Approximation . . . 172
9.6 CIRCLE_5 results . . . 178
9.7 The effect of varying the degree . . . 180
9.8 PCA plots of the error (Euclidean) in generated manifold models for $p = 2$, $g = 5$ . . . 181
9.9 Parameter space plots of the error in generated manifold model of 2DWOODBOX for $p = 2$, $g = 5$ . . . 182
9.10 Plots of the error (Euclidean) in generated manifold models for $p = 4$, $g = 5$ . . . 183
9.11 Sample images from the $p = 4$ models . . . 184
9.12 The effect of varying the degree of the curve, local interpolation test. 185
9.13 An example taken from the 4^{th} degree, local interpolation of a short stretch of STRAIGHT_5 . . . 185
9.14 Alternative knot vector for the $p = 4$ model, STRAIGHT_5 . . . 186

10.1 The use of a remainder term . . . 194

10.2 Discretisation of the domain Ω . . . 195
10.3 Visual illustrations of the 2^{nd} order 2-D scheme . . . 196
10.4 Operators for multigrid solvers . . . 201
10.5 Visual illustration of the 4^{th} order 2-D scheme . . . 202
10.6 Using ghost points at the edges . . . 203
10.7 Visual illustration of the 6^{th} order 2-D scheme . . . 205
10.8 Visual illustrations of two schemes . . . 207
10.9 Visual comparison of the results of the two different methods for the biharmonic PDE . . . 208
10.10Computational results from analytic solution . . . 209
10.11Maximum residual error vs. iterations with the biharmonic scheme 210
10.12Illustration of the experimental setup of the PDE surface model . . 214
10.13The effect of varying α . . . 217
10.142DWOODBOX, $\alpha = 10$ . . . 218
10.15Varying the number of Fourier modes used . . . 220
10.16Errors in an image produced with $N = 50$ . . . 221
10.17The effect of the remainder term . . . 221
10.18Testing the remainder term . . . 224
10.19Varying the order of the PDE . . . 225
10.20Varying the order of the PDE . . . 225
10.21The effect of varying the resolution of Ω . . . 227
10.22Comparing an example from the two solutions (interior) . . . 230
10.23Comparing an example from the two solutions (boundary) . . . 230

11.1 Linear compared to NURBS . . . 236
11.2 Linear compared to NURBS (2) . . . 237
11.3 Linear compared to NURBS (3) . . . 237
11.4 Linear, NURBS and ground truth . . . 239
11.5 Linear, NURBS and original for 4-D dataset (GAUSS) . . . 242

11.6 Linear compared to PDE 244

12.1 Maximum error in approximation, $p = 2$, $g = 5$ 254
12.2 Maximum error in approximation, $p = 1$, $g = 5$ 255
12.3 Maximum error in approximation, $p = 3$, $g = 5$ 256
12.4 Curve degree in approximation, $g = 5$ 257
12.5 Midpoint distance based curve construction, $E_{\max} = 1$, Euclidean . 260

13.1 Scatter plot of width vs. height for BRUSH 268

B.1 EXPT03 307
B.2 WINDOW3 308
B.3 KNIGHTFIGHTING 309
B.4 KNIGHTKNEELING 310
B.5 KNIGHTSTANDING 311
B.6 2DTEAPOT 312
B.7 2DWOODBOX 313
B.8 BMNOISE 314
B.9 CHESSBOARD 315
B.10 FACES 316
B.11 SINECOS 317
B.12 CIRCLE3 318
B.13 CIRCLE4 319
B.14 CIRCLE5 320
B.15 IDRISCIRCLE 321
B.16 IDRISFIG8 322
B.17 STRAIGHT1 323
B.18 STRAIGHT2 324
B.19 STRAIGHT3 325
B.20 STRAIGHT4 326

B.21 straight5 . . . 327
B.22 straight6 . . . 328
B.23 straight7 . . . 329
B.24 straight8 . . . 330
B.25 brush . . . 331
B.26 idrisstraight . . . 331
B.27 woodbox . . . 332

C.1 Circle_3 . . . 333
C.2 Circle_4 . . . 334
C.3 Circle_5 . . . 334
C.4 Idris_circle . . . 335
C.5 Idrisfig_8 . . . 335
C.6 Idris_straight . . . 336
C.7 Straight_1 . . . 336
C.8 Straight_2 . . . 337
C.9 Straight_3 . . . 337
C.10 Straight_4 . . . 338
C.11 Straight_5 . . . 338
C.12 Straight_6 . . . 339
C.13 Straight_7 . . . 339
C.14 Straight_8 . . . 340

D.1 Bmnoise . . . 342
D.2 Bmnoise . . . 343
D.3 Brush . . . 344
D.4 Brush . . . 345
D.5 Chessboard . . . 346
D.6 Circle_3 . . . 347

D.7 Circle_4 . . . 348
D.8 Circle_5 . . . 349
D.9 Expt03 . . . 350
D.10 Faces . . . 351
D.11 Faces . . . 352
D.12 Idrisfig_8 . . . 353
D.13 Idris_straight . . . 354
D.14 Knight_fighting . . . 355
D.15 Knight_fighting . . . 356
D.16 Knight_kneeling . . . 357
D.17 Knight_kneeling . . . 358
D.18 Knight_standing . . . 359
D.19 Knight_standing . . . 360
D.20 Sinecos . . . 361
D.21 Sinecos . . . 362
D.22 Straight_1 . . . 363
D.23 Straight_1 . . . 364
D.24 Straight_2 . . . 365
D.25 Straight_2 . . . 366
D.26 Straight_3 . . . 367
D.27 Straight_3 . . . 368
D.28 Straight_4 . . . 369
D.29 Straight_4 . . . 370
D.30 Straight_5 . . . 371
D.31 Straight_5 . . . 372
D.32 Straight_6 . . . 373
D.33 Straight_6 . . . 374
D.34 Straight_7 . . . 375

D.35 Straight_7 376
D.36 Straight_8 377
D.37 Straight_8 378
D.38 Woodbox 379

List of Tables

3.1 The image manifolds we study in this thesis 64

5.1 Summary of metrics 92

5.2 'Intrinsicness' results 95

5.3 Plain colour comparison results 95

10.1 Run times with the numerical schemes 209

10.2 Remainder terms at different size gaps 222

10.3 Result compared, PDE analytic, PDE numeric (taxi) 229

10.4 Result compared, PDE analytic, PDE numeric (pdiff) 229

11.1 Result compared, Linear, NURBS (taxi) 240

11.2 Result compared, Linear, NURBS (pdiff) 241

11.3 Results from the 4-D Gauss dataset 242

11.4 Result compared, Linear, PDE analytic (taxi) 245

11.5 Result compared, Linear, PDE analytic (pdiff) 245

11.6 Result compared, PDE analytic, NURBS (taxi) 247

11.7 Result compared, PDE analytic, NURBS (pdiff) 247

E.1 Linear model result summary, using taxi 381

E.2 Linear model result summary, using pdiff 383

E.3 Linear model result summary, using mi 385

E.4 Linear model result summary, using SIFT 387

E.5 Linear model result summary, using Euclidean 389

F.1 Summary of the configurations for the NURBS model 391
F.2 NURBS model result summary, using taxi 393
F.3 NURBS model result summary, using pdiff 396
F.4 NURBS model result summary, using mi 399
F.5 NURBS model result summary, using SIFT 402
F.6 NURBS model result summary, using Euclidean 404

G.1 Summary of the configurations for the PDE model (analytic) 410
G.2 Summary of the configurations for the PDE model (numeric) 414
G.3 PDE model result summary, using taxi 415
G.4 PDE model result summary, using pdiff 416
G.5 PDE model result summary, using mi 417
G.6 PDE model result summary, using SIFT 418
G.7 PDE model result summary, using Euclidean 419
G.8 pde_num model result summary, using taxi 421
G.9 pde_num model result summary, using pdiff 422
G.10 pde_num model result summary, using mi 423
G.11 pde_num model result summary, using SIFT 424
G.12 pde_num model result summary, using Euclidean 425

H.1 Result compared, PDE analytic, PDE numeric (mi) 427
H.2 Result compared, PDE analytic, PDE numeric (Euclidean) 427
H.3 Result compared, Linear, PDE analytic (mi) 429
H.4 Result compared, Linear, PDE analytic (Euclidean) 429
H.5 Result compared, Linear, PDE numeric (taxi) 431
H.6 Result compared, Linear, PDE numeric (pdiff) 431
H.7 Result compared, Linear, PDE numeric (mi) 431
H.8 Result compared, Linear, PDE numeric (Euclidean) 432

H.9 Result compared, Linear, NURBS (mi) 434
H.10 Result compared, Linear, NURBS (SIFT) 435
H.11 Result compared, Linear, NURBS (Euclidean) 435
H.12 Result compared, PDE analytic, NURBS (mi) 438
H.13 Result compared, PDE analytic, NURBS (Euclidean) 438

Chapter 1

Introduction

The appearance of an object, e.g. a vase, person's head, archeological artefact, the ruins of a castle, the environment of a robot, etc., is of great interest to a number of research areas. Primarily this is useful for a great many graphics and vision problems. The question of how to represent the appearance of objects is one which has historically received a great deal of attention by researchers from many aspects. As a result many different techniques have been proposed to solve these problems and the techniques themselves encompass a wide variety of strategies.

1.1 Context

For example more and more applications use virtual reality, whereby a real object is represented in a computer's memory and presented using various devices with the intent of providing a feel for the object which is as realistic as possible to users of the system. The system may often interact with more of the user's senses than just vision (e.g. sound, touch), however this work is focused exclusively on visual output.

Virtual environments, computers' counterpart of real environments, are typically represented using vectorial geometry: the different elements of the environment are represented using a set of shapes characterised by different parameters

such as size, position and colour. Such representations are widely used in the literature as they have a number of interesting properties. In particular, they are easy to manipulate, can provide different levels of description — from the level of simple geometrical primitives up to high level semantic descriptions — and have a good geometrical foundation.

There is however, an important trade-off between the speed and quality of the rendering of such representations. In the first instance, fast rendering is needed for convincing real-time interactivity, but good visual quality requires complex models, and results in long processing times. Furthermore accurately rendering global illumination effects is slow, even with very simple geometric models. Moreover, producing such representations is often time consuming, requiring manual production of detailed geometric representations by either surveyors or artists. Automatic methods do exist, but typically they are not very efficient and/or require many constraining assumptions. Automatic methods may, for example, only work for a specific environment (e.g. office-like) and perform poorly if used for a different kind of environment, a situation which is often encountered in real-life applications.

A solution to the problem would be to avoid using the geometry of the object to be represented and instead to use a collection of images showing all its possible appearances. Whilst this is not without its own limitations it does enable realistic environments to be rendered with a time complexity that is not directly dependent on the complexity of the scene itself.

We have already mentioned virtual environments as one major area which this thesis is connected to. This is not however the only existing work which the thesis builds upon. As we shall see in much more detail later on, this work partially unifies some aspects of existing vision and image based rendering work. It does not however focus upon the potential applications, except during the earlier stages, where we motivate our work and decisions. As well as this, the work may

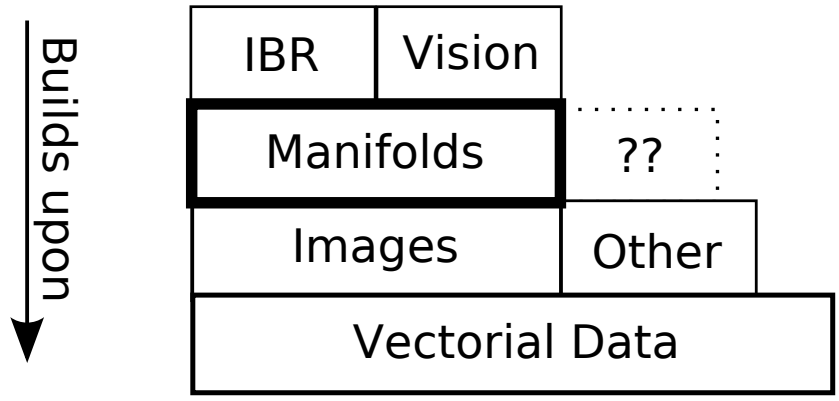


Figure 1.1: Where this thesis sits amongst related work.

potentially provide a useful framework for other classes of vectorial data, in diverse areas from genetics to audio. This view of where our work sits is illustrated in Figure 1.1, where our work, on image manifolds, is shown sitting underneath image based rendering and appearance based vision. Since we do not consider vectorial data other than images for the remainder of this thesis the corresponding block in Figure 1.1 area is shown dashed to indicate that it may be an area of future work.

1.2 Contributions

This thesis is primarily concerned with creating generic models of image manifolds and in doing so makes four main novel contributions to the literature, which are outlined at this stage.

1. **We formulate a new variant of the PDE surface method** (Ugail et al., 1999).

   This is in higher extrinsic and intrinsic dimensions than the current state of the art (the 3-D case of the numerical solution we use has been published in one noted case (Du and Qin, 2007), however we show how our solution can go far beyond this). Additionally we look at higher order variants of the PDE surface method, compared to the current state of the art (Kubiesa et al., 2004). This is important for two reasons: firstly our solution may be

directly applied to conventional existing graphical problems (e.g. a volumetric timeseries dataset). Secondly our solution makes contribution number 3 below possible.

2. **We extend existing NURBS** (Piegl and Tiller, 1997).
   We propose and discuss using NURBS in higher dimensional spaces, where the existing state of the art in the literature surveyed does not go beyond 6 extrinsic dimensions and 3 intrinsic dimensions.

3. **Modelling image manifolds using extended graphics techniques.**
   Building upon contributions 1 and 2 we propose new methods for modelling the appearance of images. These methods represent an improvement over the existing state of the art in image based rendering and manifold learning because of the generic and exact, rather than approximate, approach taken and the applicability of our methods to any and all datasets where the image manifold assumption holds.

4. **Results from large number of varied and new image manifolds datasets.**
   Finally we test our proposed methods using a number of novel datasets which are published online alongside the final version of this thesis for public use.

At this stage in the thesis the significance of these claims may not be obvious and neither have the results of our literature survey been discussed. These points will be referred to, justified and backed up at appropriate juncture and discussed in much greater detail subsequently.

1.3 Thesis outline

This thesis is structured with four main parts. The introduction, which constitutes the whole of part one, provides an overview of the field. We review literature

which is generally relevant to the whole of this thesis in Chapters 2 and 3, and show how this thesis fits within the existing body of work. Further we clearly show through this review how our work differs from existing work. The second of these two chapters then proceeds to extract and introduce a core concept, which we term "image manifold" and show how this concept relates to a diverse range of existing literature, as well as providing some examples of image manifolds we will be studying in more depth. In the final two chapters of the introduction (4 and 5) we set out the research question and the aims and objectives tied to its answering. Finally we address several key issues with our methodology, establishing a principled approach for evaluating our work.

The literature review in the introduction draws from many diverse fields of study. In order to provide both structure and ordering to the review we have categorised each and every item reviewed. A summary of these are provided in Appendix A.

Many other works which we draw from are covered as and when it is topical in order that may be directly tied to specific chapters. Topics are therefore only reviewed when they become pertinent.

In the second part of this thesis we build further on the notion of an image manifold by studying in depth the properties of the many examples we introduced. We address in turn two key aspects of working with image manifolds. Firstly we consider two measurable local properties of our image manifolds, and discuss the implications of our observations. Secondly we look at the notion of dimensionality from a theoretical standpoint as well as applying it to our own image manifolds. The issue of dimensionality has important implications for the remainder of this thesis and we discuss these further also.

In the third part we draw upon our observations and conclusions from the previous two parts to produce three numerical models of the image manifolds themselves. We do this using two key approaches, NURBS and PDE surfaces,

with a simple linear model introduced by way of comparison. For both of the main techniques we discuss both theoretical and practical aspects of their implementation and use within this context, as well as applying them to the example image manifolds we have studied previously. For each implementation we evaluate its effectiveness both visually and numerically, although a fuller evaluation is delayed until it can be presented in the broader context later on.

Finally in the fourth part of the thesis we seek to unify the various strands of the thesis. In Chapter 11 we follow up with an in-depth analysis and discussion of the results we have presented previously. In Chapter 12 we show how our measurements from Chapter 6 can potentially enable us to produce better models of image manifolds. Finally we concluded with Chapter 13, which looks at what our results mean for our hypothesis, and looks towards the implications and potential future work which arise out of this thesis.

The results of our experiments produced a huge volume of data, which we have summarised in several ways. To address this, throughout our discussions of the results, we pick out 'key' examples. These examples are either unique and interesting in someway, or representative of the general trend which can be observed in the results. We note this as appropriate, however visualisations and complete, more detailed results are provided in the attached appendices to permit the interested reader to verify our claims for themselves.

Part I

Foundation

Chapter 2

Image Based Rendering Literature review

In this chapter we introduce one area of work relevant to our own, and review some of the existing literature. This work is only one of the relevant areas, and so in Chapter 3 we introduce another as well as discussing some more recent advances in this area. A summary of literature reviewed is provided in Appendix A.

2.1 Introduction

Traditionally in computer graphics images are rendered from a collection of geometric primitives, typically triangles, and often a set of properties that influence how they are displayed. The computer generated characters we see on screen in many recent films are typically constructed using at least 10^7 polygons. This approach to graphics tends to be very time consuming for artists who must work almost at the level of individual polygons. Furthermore global illumination effects are extremely time consuming to render with all but the most simple models.

There is another approach being taken by some in the graphics community, which has steadily been growing in both frequency of use and usefulness. The basic premise of this is that instead of using geometry as an input to render images, a

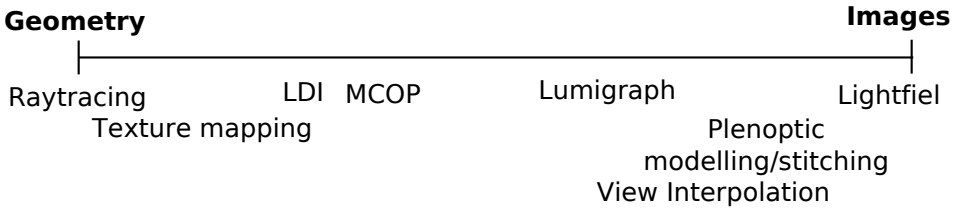


Figure 2.1: The continuum of rendering techniques ranging from purely IBR to exclusively geometry dependant. Based on (Shum and Kang, 2000, Figure 1)

set of images themselves form the input to the rendering process. This concept is known as Image Based Rendering, henceforth referred to as *IBR*.

In this chapter we review a number of the existing approaches to image based rendering, which are loosely termed 'classic' IBR. In subsequent chapters we shall expand on this to look at more recent developments in IBR as well as view synthesis which arises out of other areas of research. There are already a number of reviews of this area. Throughout this chapter we adopt the structures and terminologies from (Shade, 2002; Shum and Kang, 2000). These two reviews provide a framework in which existing image based rendering techniques may be classified.

These two approaches, of image based rendering and geometric models, are not mutually exclusive however. In fact we can consider these approaches to be the extremities of a continuum (illustrated in Figure 2.1), ranging from using no geometry at all to using exclusively geometry to render images. More often than not techniques that rely upon geometry are augmented with a technique known as texture mapping, whereby geometric primitives have an image (texture) superimposed on them as they are drawn.

All IBR algorithms, which are points on the continuum, can be loosely grouped into one of three categories, namely those which require no geometry, those requiring geometry and a third, middle category. Members of this third category typically use something like a feature correspondence across image sets, in addition to the images themselves as an input, and thus are using some, albeit simple, geometric information for their rendering.

Section 2.2 introduces an important background function which provides a frame of reference for comparing IBR techniques. The remainder then presents an overview and discussion of a selection of common IBR techniques. An understanding of at least the common elements of these techniques is important to realise the potential of other ideas discussed later. Much of the literature reviewed in this section is also reviewed in (Shade, 2002; Shum and Kang, 2000), and our categorisations are based upon these.

2.2 The plenoptic function

One of the fundamental concepts in IBR is the notion of the plenoptic function which was formally stated in (Adelson and Bergen, 1991). Interestingly this was originally presented as work in the area of vision, yet it has found heavy use here in the context of a graphics problem. The plenoptic function is defined as:

$$P = P(\theta, \phi, \lambda, t, V_x, V_y, V_z). \tag{2.1}$$

P is a seemingly complex function describing all rays of light passing through the lens of a hypothetical eye with a 360° field of vision in 3-D at (V_x, V_y, V_z). P is the intensity distribution for a given wavelength λ, at a given time t, arriving at the hypothetical eye from a given direction θ, ϕ. This is an enormous amount of information to even consider beginning to sample, but by fixing certain parameters, e.g. t, V_x, V_y, V_z, we see a familiar construct, namely a regular 2-D photograph.

As an example, given in (Shade, 2002), the amount of data required to sample the plenoptic function within a finite domain, namely that of a $7m \times 5m$ office at a fixed point in time, requires approximately 26GB of space, given a compression ratio of 20:1 in all the sample images. Even this example is not a complete sampling within the finite domain.

Often it is convenient to replace θ, ϕ with x, y as coordinates on an imaginary

plane located a fixed distance from the 'eye', thus the function becomes:

$$P = P(x, y, \lambda, t, V_x, V_y, V_z). \tag{2.2}$$

Almost all the IBR techniques discussed henceforth can be thought of as in some way reducing and representing a part of this plenoptic function (Shade, 2002), typically in some structured fashion so as to permit lookups. Generally the sampling strategy is driven by a target quality for output images, although this varies from technique to technique and will be discussed on a case by case basis in the remainder of this section.

Being a 7-D function, the plenoptic function is inherently difficult for human beings to visualise. As mentioned previously however by constraining several of the dimensions we can begin to understand *slices* of it. Figure 2.2 shows a number of possible slices from a simple example.

Shade (2002) outlines four different techniques which IBR algorithms typically use to approximate the plenoptic function. These are: "*Taking a subset of the plenoptic function*", by which they mean restricting the number of samples to reduce the otherwise infinite domain of the plenoptic function; "*Taking a slice of the plenoptic function*", since reducing the total number of degrees of freedom can make large reductions in the amount of data required, which typically can be positional or view orientation restrictions; "*Sparsely sampling a parameter*" is similar to slicing, however it still retains some of the more pertinent information that would have otherwise been lost through sampling; "*Eliminating redundancy*", any single sample image will typically contain pixels that are visible from many of the other samples, sharing this information instead of duplicating it can lead to large savings.

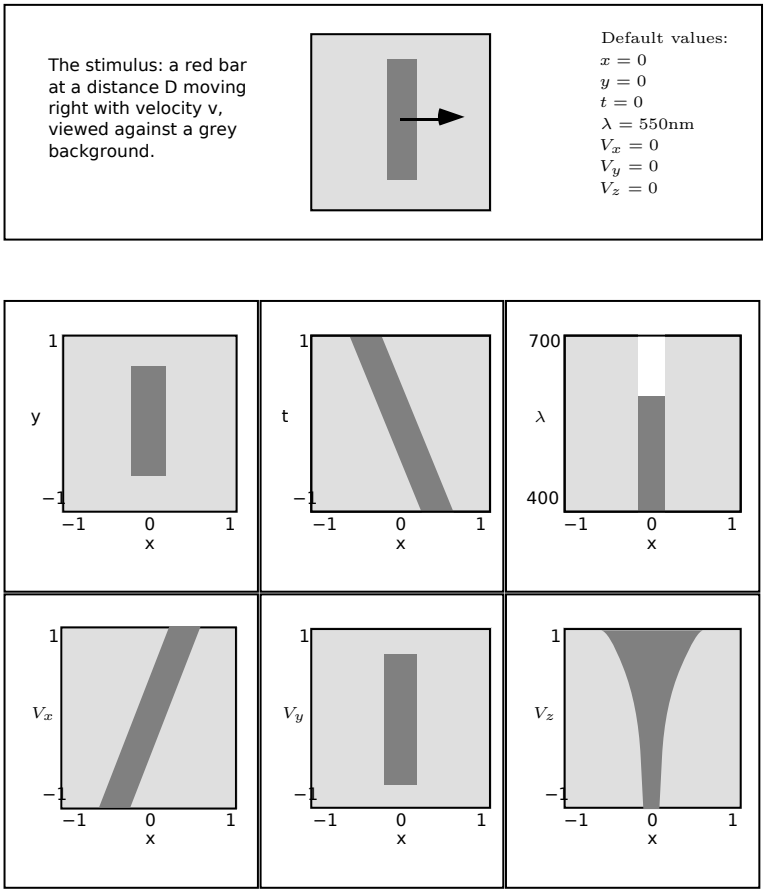


Figure 2.2: Small slices of the plenoptic function, image from (Adelson and Bergen, 1991)

2.3 Explicit Geometry

IBR techniques categorised as using explicit geometry typically require depth maps associated with every input image. One could also consider texture mapping to be an extreme form of IBR which uses explicit geometry. The examples reviewed here are less explicit in their use of geometric information however.

2.3.1 MCOP

Multiple centre of projection, or *MCOP* (Rademacher and Bishop, 1998), is an IBR technique which is surprisingly reminiscent of paintings by Renaissance artists, who used multiple vanishing points as a technique to direct viewers' attention towards specific features in a painting.

Figure 2.3: Example MCOP image, from (Rademacher and Bishop, 1998)

In this case however MCOP is a technique for representing images, typically of an object, in such a way as to allow re-projection of many normal (i.e. single centre of projection) images from a single MCOP image. Obviously this re-projection requires depth information which, like other IBR techniques that use depth information, tends to be a by-product of 3-D rendering processes. This information is not so easily obtained when using a real world CCD setup, and requires additional hardware such as a 1-D laser range scanner to be calibrated and operated synchronously.

The acquisition of real world MCOP images is further inhibited by the fact that no real camera exists capable of capturing an MCOP image at a single instance. Instead, typically a camera is moved around an object slowly, repeatedly taking images and only a specific column of each is considered. This means that, like many other IBR techniques discussed, the sampling assumes a static environment and temporal incoherence may be observed if this is not the case.

In the example MCOP images given in Figure 2.3 each column is captured with a different centre of projection from that of its neighbours. That is to say the x coordinate of a pixel relates to a specific centre of projection. The y coordinate of a pixel does not change the centre of projection of the camera.

This setup has several advantages. Firstly in the case of sampling convex objects this provides uniform coverage of the object as it is rotated, whereas if views of the object were captured via a small number of conventional images

some areas of the object would be sampled at a greater distance from the camera than others. Secondly if the change in centre of projection changes in a simple way proportional to the x coordinate then re-projection of the MCOP image to a given view is simplified.

It is possible however to take a more complex path through a scene than just a single rotation. In this case information about the trajectory of the camera must be stored explicitly besides the image, whereas previously it was implicit. Furthermore it is possible that a specified camera path will result in some areas being multiply sampled. In this case the rendering of the re-projection may need some additional intelligence to ensure the best sample is selected, rather than simply the one rendered last.

View reconstructions from MCOP images do not truly reflect view dependant lighting unless there are multiple samples for each point, and the correct one is selected somehow.

MCOP relates to the plenoptic function in that it is a series of vertical slices through an image.

2.3.2 Layered Depth Images

Layered depth images (Shade et al., 1998), referred to as *LDI*s henceforth, is a method used to produce an effect of parallax, which is missing from a static photograph. This technique provides an illusion of parallax by using *sprites* associated with depth information for each pixel. This additional geometric information permits subtle changes to a visible scene as an observer moves small distances in it.

Take for example a scene made up of a tree with many leaves. If the tree was constructed using a traditional sprite the leaves within the tree would not display parallax, as a viewer would expect and would accordingly appear unrealistic because of this. By adding depth information, and several layers to each sprite it is

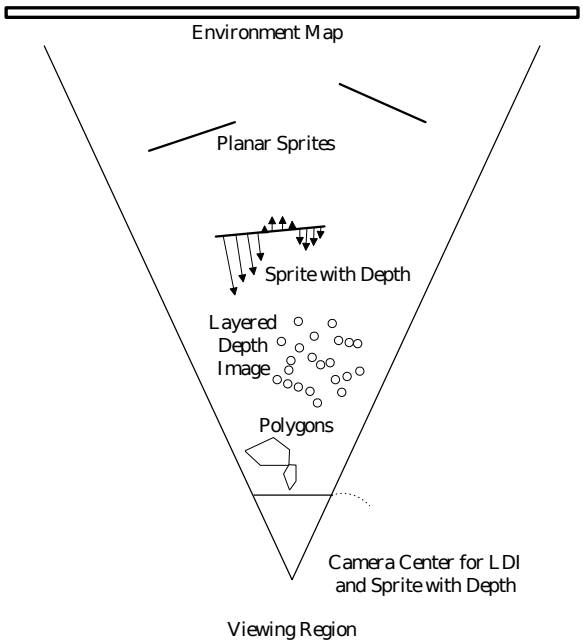


Figure 2.4: Example scene using LDI, image from (Shade et al., 1998)

possible to overcome this.

This can work well in practice for specific scenes. It does not consider effects such as specular highlights which will vary based upon the viewer's position in addition to the object itself. Acquisition of LDIs from the real world becomes a traditional computer vision problem. Generating artificial LDIs however is relatively straightforward using a modified raytracer. Furthermore, the choice of primitive varies greatly depending upon the depth of the object it represents in the scene, and the type of the object itself. This increases the complexity required for generating a scene to such a point that it is not far removed from the physically based models of a scene.

2.4 Implicit Geometry

IBR techniques considered as using implicit geometry are generally techniques which rely upon some form of feature correspondence in images, for example knowledge of the locations of cameras or identification of the same feature in two images via some other method. These methods are often very similar in form

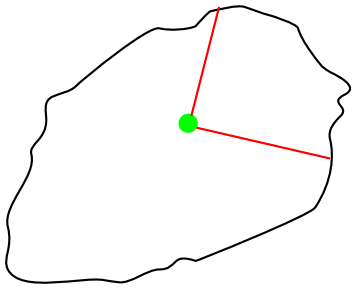


Figure 2.5: Plenoptic stitching parametrisation example

to the classical stereo correspondence problem. This similarity serves to emphasise further the overlap between Vision and Graphics that runs through much of this work. Implicit geometry methods typically is used to make simplifying assumptions and imposes constraints that permit solving an inverse problem that is normally extremely hard.

2.4.1 Plenoptic stitching

Plenoptic stitching (Aliaga and Carlbom, 2001) proposes to use samples of the environment, obtained from closed image loop, formed of omni-directional images. By selecting and combining pixels from these samples, new views from within the loop may be constructed. Figure 2.5 shows an example for a viewer (green) where an image is constructed, column by column for a given view (red lines represent frustum), which is made of pixels from the images around the loop (grey boundary).

Exactly which column comes from which image is selected based upon camera pose estimations which are recorded along with each sample image. This clearly means that good results from this method are dependant on good pose estimation. The authors claim an average pose error of 0.5% in their example from within an office. The pose estimation technique they use is based upon a post with lights on it. This is potentially quite intrusive in the environment, and in many other scenarios could be completely impractical.

2.4.2 Image-based priors

Another geometry-free approach to view synthesis is presented in (Fitzgibbon et al., 2005). In this work the authors use statistics of the images as a constraint to control the new images which are synthesised. In this way they ensure that generated images appear more plausible than they otherwise would. The idea with this work is that prior knowledge of a scene required to constrain the image can be learnt from real images.

At each pixel in the synthesised image the authors estimate the colour which is most likely to be a re-projection of part of a 3D object, based upon the other views of the object and the location of the camera. This estimation is performed using Bayes' rule.

This method constrains the output solution, which addresses a typical problem seen in many IBR algorithms where the synthesised view is in some way composited from a number of samples, and where those samples mix they do not match up. This normally takes the form of 'ghosting' or 'echos' withing the synthesised view. The method assumes that the view is of a three dimensional object, with 'normal' image statistics. This is not a problem in a great number of cases, but there are nonetheless many instances where this may present a problem.

2.4.3 View interpolation

View interpolation (Chen and Williams, 1993) exploits the fact that as a camera moves, the amount objects visible to the camera appear to move is dependent upon the distance of the object from the camera. Using the known camera locations, an 'offset vector' can be calculated for each pixel in the image. For range images, or computer rendered images this information can truly be computed per pixel, however for other images estimating the information is time consuming and error prone.

Novel views in between a pair of images are computed using linear interpola-

tion, moving the pixels based upon the offset vectors however. Occlusions in the images may introduce holes in the resulting output images. Even when there is no occlusion what was one single pixel in one view may well not correspond exactly with just one pixel in the output view after perspective projections are taken into account.

2.5 No geometry

This section discusses IBR techniques that do not rely upon geometric information in their representations of an environment.

2.5.1 Lightfield and Lumigraph

Lightfields (Levoy and Hanrahan, 1996), and Lumigraphs (Gortler et al., 1996) are two techniques which were co-developed, yet very similar overall. These are good examples of IBR techniques. They can be considered an array of perspective images of an object taken from nearby viewpoints within the same plane. Essentially lightfields work by noticing that, provided there are no occlusions the view from a point within a given space can be represented in terms of a small number of samples of that space.

Assuming sufficient samples then the images between them contain information about all light entering any given point within the Lightfield. In this way, by careful selection of pixels across the collection of images, it is possible to create images at an arbitrary position anywhere within, or even behind the Lightfield.

The interesting simplification made in both Lightfield and Lumigraph is the drop from a 5-D (since t and λ aren't varying to begin with) to a 4-D plenoptic function. The rays of light are thought of as passing through two planes, which define a bounding box for a scene, thus the function becomes $P = P(u, v, s, t)$, where (u, v) and (s, t) define points in the respective planes.

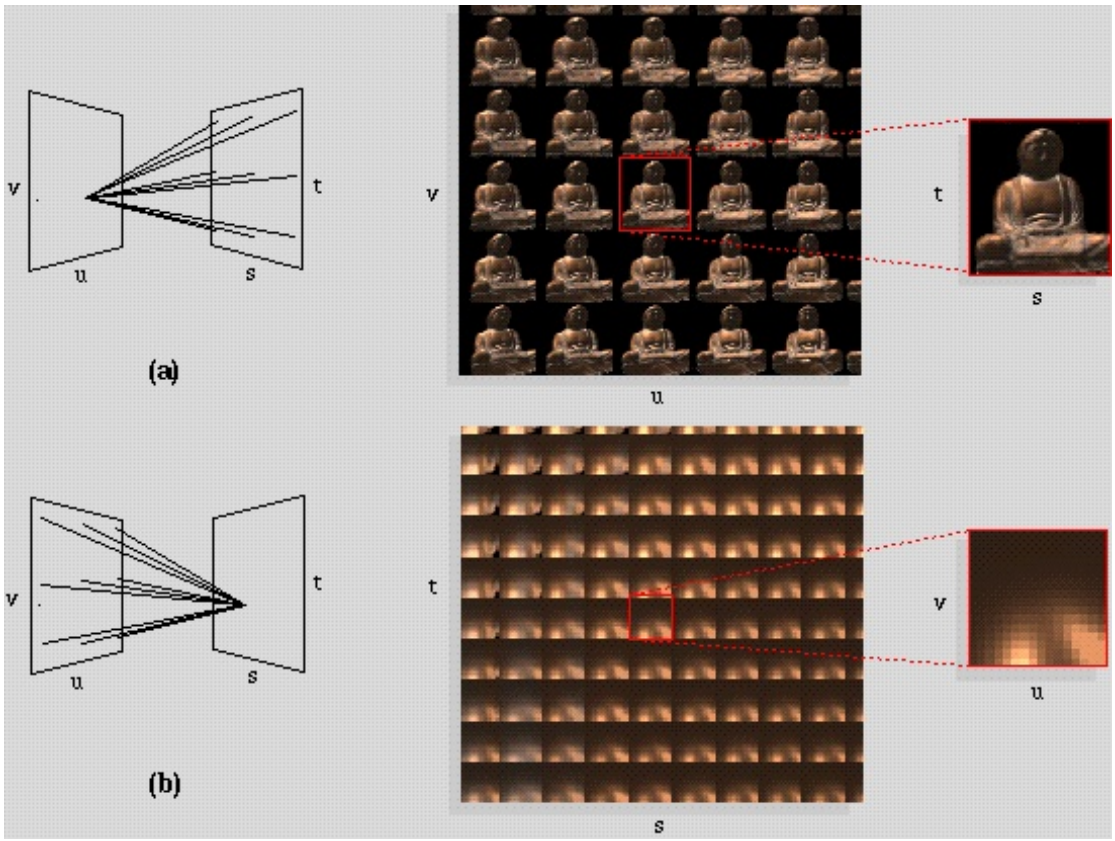


Figure 2.6: Example of lightfields, from (Levoy and Hanrahan, 1996)

This simplification is permissible because of the assumption that the space within which the viewer is permitted to move is "free space", i.e. there are no objects within it, which would otherwise affect the passage of a ray of light through it.

Both Lightfield and Lumigraph make significant compression via the use of a quantisation scheme, which allows for random access still.

Using Lightfields with medium-low sampling implicitly assumes that the depth of the subject of the image is constant. However, Lumigraph uses a depth-correction technique to try and overcome this, but this depth-correction requires depth information as with the other implicit geometry IBR techniques[1]. In (Isaksen et al., 2000) the authors have attempted to address these weaknesses by the

[1]Indeed it should really be discussed in the Implicit Geometry section, however the many similarities between Lightfields and Lumigraphs warrant discussing them both together, and "No geometry" was selected rather arbitrarily.

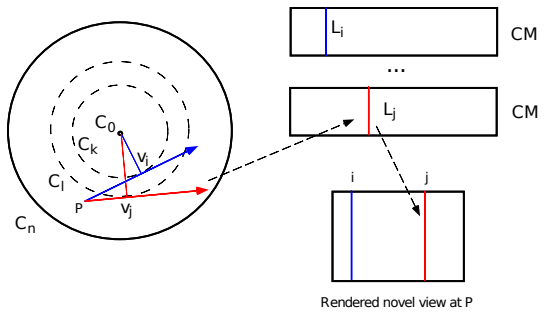


Figure 2.7: Concentric mosaics, figure from (Shum and He, 1999)

addition of multiple moderately sampled focal planes for each ray. This enables dynamic control of things such as depth of field and focus when rendering new views, much as one would with a traditional camera.

These techniques are somewhat constrained by the free-space assumption, which would make modelling large real-world environments impractical. The construction of lightfields also requires specialist equipment.

2.5.2 Concentric Mosaics

Concentric Mosaics (Shum and He, 1999) sample the plenoptic function in 3 dimensions assuming a viewing camera constrained to exist within a system of planar concentric circles. The result of this is that synthesised images only appear to have horizontal parallax.

This is somewhat more convincing than just a simple 2-D panorama, yet not enormously more data. The requirements to capture a concentric mosaic of a scene are probably the lowest of any of the techniques we review here. All that is required is the ability to rotate a regular camera around a fixed point at various radii.

Novel views are created by compositing rays from the existing view which match, the required ray in the new view. This is possible because the 'free-space' assumption is used here again.

2.6 Conclusion

This chapter has provided an overview of what we termed 'classic' IBR algorithms, which have been gaining momentum over purely geometric based approaches.

What we have seen here is a number of examples of image based rendering algorithms. The algorithms presented here all share one common feature: based upon existing images of some description and usually some implicit or explicit knowledge of the geometry of a scene a (limited) set of novel views may be synthesised by using this knowledge to appropriately select or combine pixels from existing images. The interesting thing to note here however is that each and everyone of these algorithms makes simplifying assumptions to reduce the dimensionality of the function being approximated. This has several important implications for their usage. Primarily this means that the usage is constrained in some way, e.g. viewing from inside the convex hull of an object. Additionally though this means that the number of situations in which the algorithm may be used is constrained.

In comparing them with more traditional geometry based algorithms it becomes clear that unlike with geometry based rendering time complexity is not a function of scene complexity. Consider for instance the Lightfields algorithm (Section 2.5.1): whether or not the object it is rendering is a simple cube or an ornate statue that would require millions of polygons to capture all the details the run times do not change. Instead, though, the size of the input and output images (and the algorithm design obviously) are the main factors affecting computation times.

In addition to this, using real world photographs as inputs to IBR algorithms results in implicit sampling and use of properties of objects such as reflectance which are hard to automatically extract and reproduce. As well as this, many of the IBR algorithms discussed do not require any specialist equipment to capture an object or scene.

All of the IBR techniques we have surveyed here introduce significant restric-

tions in one form or another on either the type and nature of the objects or environments they can model, or on the quantity and location of the views a viewer can generate using them. If the goal of image based rendering is to model the plenoptic function in the general case, then these offerings all fall short and IBR remains an open problem.

There have been a number of references made to the advantages of IBR algorithms, however IBR is not without its disadvantages. One major disadvantage of such algorithms however comes directly from working at a sub-symbolic level. In all the representations there is no direct concept of an 'object', nor is there any immediately obvious hierarchy of objects. This makes manipulation of inputs to IBR algorithms a particularly difficult problem. For instance editing an existing scene, so as to place a teapot on a table using IBR techniques is not possible directly on such a symbolic level. There is some work however looking at automatically removing areas from images, and replacing them with synthetic textures, for instance that of (Alotaibi, 2009; Efros and Leung, 1999; Labrosse, 2003; Wei and Levoy, 2000). This is an area of significant interest, and much active research, likely to yield improvements in the future.

The relevance of image based rendering to our own work is two fold. Firstly one of the potential applications of this thesis is a novel image based rendering technique. This influences some of the decisions taken later on, for example that of introducing the perceptual evaluation metric (Chapter 5) which we subsequently use to evaluate all our experiments. This metric was selected with IBR as a potential application in mind. Secondly, the plenoptic function itself, which was originally intended to express "elements of early vision" has been used to unify the IBR techniques we have discussed. This is an example of the convergence of vision and graphics, a theme which runs right through the core of this thesis.

This is not the end of our discussion of IBR, which is a major field. In the next chapter we will introduce several further concepts which have facilitated more

recent developments in IBR.

2.7 Summary

In this chapter we have seen:

- an overview of existing image based rendering techniques;
- a framework that connects and relates all these techniques.

In the next chapter we will explore:

- how image based rendering and appearance based vision are connected through the concept of *image space*;
- dimensionality reduction;
- further examples of IBR which build upon these concepts;
- some examples of appearance based vision;
- the datasets we will be considering throughout this thesis.

Chapter 3

Image space and image manifolds

All of the literature reviewed in this chapter and the previous can be found in a copy of our summary notes attached to this thesis in Appendix A.

3.1 Introducing image space and image manifolds

An image $\mathcal{I}$ can be considered a point in a $\mathbb{R}^{w \times h \times c}$ space spanning all images of width w, height h, and having c colour components per pixel. This space is often referred to as *image space*, e.g. (Nayar et al., 1996).

A manifold is defined in (Spivak, 1979) as "a metric space[1], M, where if $x \in M$ then there exists some neighbourhood, U of x and some integer $n \geq 0$ such that U is homeomorphic to $\mathbb{R}^n$". Zomorodian et al. (2005) define a "*homeomorphism* $f : \mathbb{X} \to \mathbb{Y}$ [as] a $1-1$ onto function, such that both f, f^{-1} are continuous". In practice this means that anything which is a manifold can, locally at least, be treated as Euclidean, even if globally this is not the case, and that some mapping between these two spaces exists.

We can apply this concept of a manifold to a specific case, namely *image*

[1]Strictly we should discuss topological spaces instead of just metric spaces, however since all metric spaces are topological spaces this simplification is justifiable here.

manifolds, which are collections of related images, varying between them in some parameters which control the acquisition of the images in the collection (of all the sampled images). This is not novel, and we are not the first to introduce this term, see for example (Lu, 1998) (note that Lu (1998) uses the term "pixel space" to describe what we term "image space"). The parameters controlling the manifold could, for example, include time, if the appearance varies over time, or it could be a rotation, translation or general transformation of real world objects that are captured by the images, or the camera itself. Some examples of image manifolds are presented and discussed further in Section 3.4.

The concept of *image manifolds* is one which has become increasingly common in computer vision (e.g. Bichsel and Pentland (1994); Donoho and Grimes (2005); Lu et al. (1998)). This is made possible by considering images as points in the high-dimensional image space. Images of an object undergoing a series of transformations are often not randomly placed in image space. Instead, under certain circumstances, these resulting images can be approximated as n-manifolds, where n is the number of independent variables controlling the image transformations. This mapping between the high-dimensional image space and a lower dimensional space has been particularly useful for solving a number of vision problems.

This concept has been applied in solutions to a wide variety of problems, the most notable being face recognition (e.g. Bichsel and Pentland (1994); Moghaddam (1999); Tenenbaum (1998); Turk and Pentland (1991*a*)), although it has also been applied to other vision problems (e.g. Labrosse (2007); Murase and Nayar (1995); Nayar et al. (1996); Neal and Labrosse (2004)).

Recently work has been progressing on the idea of using image manifolds to capture the appearance of an object for the purpose of synthesising other views (e.g. Shum et al. (2002); Tenenbaum (1998)). Tenenbaum suggests that image manifolds could be learnt to enable construction or recognition of new views which "can then be carried out by linear operations in the feature space". Other ideas

include image compression, such as the "Statistical Manifold Coding" (SMC) presented in (Lu, 1998), which besides SMC provides an in depth discussion of many other image manifold related issues.

Previous work has looked at some properties of image manifolds and their topology. This includes (Bichsel and Pentland, 1994; Donoho and Grimes, 2005; Wakin et al., 2005), and perhaps most significantly that of Lu (1998). Presented in (Donoho and Grimes, 2005) is a theoretical study of the ISOMAP (Tenenbaum et al., 2000) procedure, which has been used to try to automatically extract the structure of a manifold. Here however, we are not concerned with extracting the structure of an image manifold since this is already inherently known, in our case, from the method used to sample the image manifold in the first place.

3.2 Image space IBR

In addition to the IBR literature we reviewed in Chapter 2 there are a number of examples of more recent IBR work, which build upon the concept of image space more directly. In this section we will review and discuss a number of these methods.

Video Textures (Schödl et al., 2000), where a sequence of input images is used to synthesise new videos, have been applied to solve a number of IBR problems. This includes generating continuous infinitely varying output streams of images, and streams of images suitable for looping. The concept of video textures has been generalised (Phillips and Watson, 2003) to cover audio as well as video. The central concept of video textures is that input images are compared to each other (using, e.g., a distance metric in image space). Once the full distance matrix for all of the available input images has been computed a suitable frame to show may be selected simply by choosing a similar frame to the current one and optionally using an heuristic such as morphing to smooth the transition. In some cases similar (or even identical) images may be found in a video sequence which do not

make good candidates for a transition. This tends to arise when the dynamics of the system being recorded are such that the same image can be observed multiple times when the underlying system is in a very different state. The authors address this problem, and give an example of such a problem by way of a pendulum swinging. Their proposed solution is to take not just the similarity of individual frames themselves, but of the close neighbour, which will successfully penalise false matches from the pendulum example. Another potential problem which the authors address is that of 'dead ends', where transitioning to a frame results in a situation whereby the possible choices for subsequent frames is very small, or even non existent. To avoid this they propose another variation on the basic cost function, which includes the cost of future transitions as well. The primary advantage of this method is that by reusing selected input frames in this way the output is almost guaranteed to be visually plausible and the cost of synthesising new frames is relatively low (i.e. just a morph) once the distance matrix has been computed. This method is not well suited to scenes which are not cyclic or periodic in some way, because the cost of jumping backwards in time will always remain relatively high. More recently a variant has been proposed (Agarwala et al., 2005) which generates panoramic video textures from the panning of a standard tripod mounted camera. Here the authors seek to break a scene into static and dynamic parts. The static parts are registered and handled as a typical panoramic scene, whilst the dynamic parts are split into patches and an approach similar to the basic video textures method is used for selecting how to transition between frames. Like video textures themselves, this directly re-uses input patches from the video, which ensures that the output maintains a high level of visual plausibility. One interesting side effect of dividing the panorama into patches is that a given view of the panorama is likely to contain patches of the video sampled at different points in time, and thus potentially introducing temporal incoherence within a single output view.

Another approach to IBR has seen many authors taking a statistical approach to modelling textures (Bar-Joseph et al., 2001; Szummer and Picard, 1996; Zhou et al., 2009). One particular example of this, dynamic textures (Doretto et al., 2003), seeks to learn and extrapolate images which are the output of some dynamical system (e.g. smoke, fire, gaits). The motivation for, and main advantage of, this work has been to avoid some of the limitations of other physically based dynamic texture synthesis algorithms, where simulation of a physical model, derived from first principles, is used to synthesise new images. Primarily the limitations with these physically based methods arise from their close connection with particular sub-classes of dynamic textures, although often these models have high computational overheads as well. By learning the parameters of a first order autoregressive moving average (ARMA) model the "essence" of the dynamic texture itself is learnt. This amounts to a system identification problem (see (Costantini et al., 2008) for another approach to solving this in the IBR context, using Tucker decomposition). The resulting model is generative, relatively compact and suitable for use with recognition tasks as well. By changing the model parameters it is possible to manipulate the dynamics of the system and alter the output.

Recently phase-space embedding (Basharat and Shah, 2009) has been proposed as a solution to IBR problems. By modelling the underlying dynamical system which was observed in an input time series (e.g. images) in a way that permits predictions in phase space, and hence reconstruction of a time series, new time series maybe synthesised. Ideally, the mapping function used to make predictions in phase space would be selected based upon known properties of the underlying system, however in the general case this is not known. Instead the authors propose a mapping function based upon a weighted average of neighbours in phase space. To estimate the dimensionality of the phase space, the authors recommend a method based upon reducing the number of false nearest neighbours generated by the embedding. Working in phase space has the advantage of potentially re-

ducing quite significantly the dimensionality of the space in which the predictions are made, which in turn can keep the cost of predictions low, as well as facilitating visualisations of the strange attractor. The authors propose both univariate and multivariate phase space reconstructions of the inputs, which they applied to both human motion and dynamic textures with better results than a number of similar methods. Like other dynamic texture methods this method makes strong assumptions about the nature of the underlying data, which makes the results suitable for only a subset of IBR problems.

Other approaches to dynamic texture synthesis have sought to create image based models (as opposed to physical models) of the particles which make up a scene. For example (Wang and Zhu, 2002, 2004) use linear combinations of Laplacian of Gaussians, Gabor or Fourier bases to represent elements of textures and construct model the dynamics of moving elements (which they call "movetons") of the scene such as snowflakes, raindrops, birds in flight, etc., modelling birth (source) and death (sink) as well as the trajectories of particles within the scene.

Souvenir et al. (2006) take an entirely different approach again, using B-Splines to model the deformation field in the image plane of images of a heart. The modelling is formulated as a minimisation problem, and evaluated using real patient data as well as a synthetic dataset. The evaluation is conducted by removing some of the input images, synthesising them and comparing to the known ground truth images. This works well for their specific problem domain, because the assumption that the changes between a reference image and any subsequent image of the same heart are a deformation that fits well with the physiological realities (deformation due to breath cycle and heartbeat itself) of MRI scans of a patient's heart.

3.3 Appearance based vision

In this section we review a number of existing works, which either implicitly or explicitly build upon the concepts of image space and image manifolds. These

examples are important to us because they offer some initial evidence to support the idea of image manifolds, through the positive results they present.

One of the key problems with traditional, feature based vision methods is the development of truly generic algorithms for solving problems. In a typical vision algorithm the images are pre-processed in some way to extract features. These might be edges, corners, blobs, or even more domain specific features. Often an algorithm that works well with one problem will not transfer well to another, seemingly similar problem, simply because the feature extraction phase fails.

3.3.1 Robot vision

In (Nayar et al., 1996) the authors suggest a method of avoiding comparing a new image to every single previously seen image when trying to recognise an unknown input image. This would be computationally very expensive if the number of previously seen images is large. Furthermore, the input image may not correspond exactly with any one previously seen image. The authors instead propose a binary search through the high-dimensional space to identify the closest manifold point. This work is one of the key works in terms of using a sub-symbolic (i.e. purely appearance based) approach to working with robot vision.

In the discussion in (Nayar et al., 1996) many arguments we have previously mentioned are covered. This includes the problems associated with feature extraction which would otherwise be required. Additionally they point out that such a technique is not limited to just RGB images, but would also work with range images for instance. Another advantage of their proposed method is the non-reliance on shape and reflectance properties of objects. These can vary greatly between objects, and can be partially responsible for the failure of a feature detection algorithm on a different class of input images. They also suggest the use of non-uniform sampling to allow fine control of positioning in areas which require it. This suggestion is particularly interesting because it is one that could be po-

tentially exploited further in this work and is discussed briefly in the aims and objectives section of this dissertation (Chapter 4). The implementation in (Nayar et al., 1996) ran in real time on commodity hardware, which was relatively novel for robot vision at the time.

In (Neal and Labrosse, 2004) the authors propose modelling the environment of a mobile robot using a set of connected images. These are processed using an algorithm that is based on a model of immune system memory. In this way the immune system memory model can be viewed as a dimensionality reduction technique. This allows the method to model the topology of large environments compactly, but does not require a full model of the entire image manifold associated with the images of the environment. The artificial immune system algorithm used is very sensitive to a number of parameters associated with it — poor choices of values can lead to the formation of either several, disjointed graphs, or the inclusion of too many images. The resulting model of the appearance of the environment, whilst suitable for navigation style problems is not suitable for use with IBR type problems, because of the sparsity of the retained samples.

Similarly in (Labrosse, 2006) a method is proposed to estimate the change of heading in a mobile robot fitted with a panoramic camera based solely upon the images from the camera. By making planar shifts to 'unwrapped' images, and comparing to a previous, known orientation it is possible to make good estimates of the total rotation. The precise amount of shift required to get the best match is indicative of the amount of rotation the robot has performed between the two images, provided the robot has only rotated around the optical axis of the camera between the images. In instances where the robot has moved by a sufficiently small amount this can still provide good estimations of the rotation that has occurred. This clearly isn't fool-proof, and it is possible to construct hypothetical scenarios which would cause this to fail, but in the majority of real-world environments there exists a global minimum that corresponds at least as accurately with the amount

of rotation as a magnetic compass or inertial navigation system. This algorithm relies upon the closeness of two similar images (but unlikely to be identical due to real-world sampling, small movements, etc.) in image space. We will consider this method further in Chapter 5, when we look at the concept of distances in image space in more detail.

Proposed in (Sturzl and Mallot, 2006) is an alternative to measuring distances between images in image space for sub-symbolic visual homing. Instead of trying to calculate the home vector based upon distances in image space it is proposed to measure distance in image frequency space. The concept of working in image frequency space is one which we do not consider further in this thesis, due to time and space constraints, but may well make an interesting area of future work.

Finally, in (Labrosse, 2007) the author proposes a method for estimating local movements on an image manifold, in order to aid the performance of short range visual homing. This method is based upon the projections of the omni-directional camera, but the mere fact that this method is capable of producing useful approximations of small movements is of interest to us here; it would seem to indicate that the images involved are well structured, and locally predictable, an important property for our work.

3.3.2 Faces

Principle component analysis, or *PCA*, is a technique for dimensionality reduction. PCA is sometimes referred to as the discrete Karhunen-Loève transform. As such it has been used in the past to perform dimensionality reduction required for face recognition, such as that of (Turk and Pentland, 1991*a*,*b*). In this way any face image can be projected into "face space", i.e. it can be expressed as a linear combination of principal components. Turk and Pentland (1991*a*) refer to the axis of this face space as eigenfaces, or simply the eigen vectors of the set of faces. Clearly these features do not have to correspond to ears, eyes or any other part of

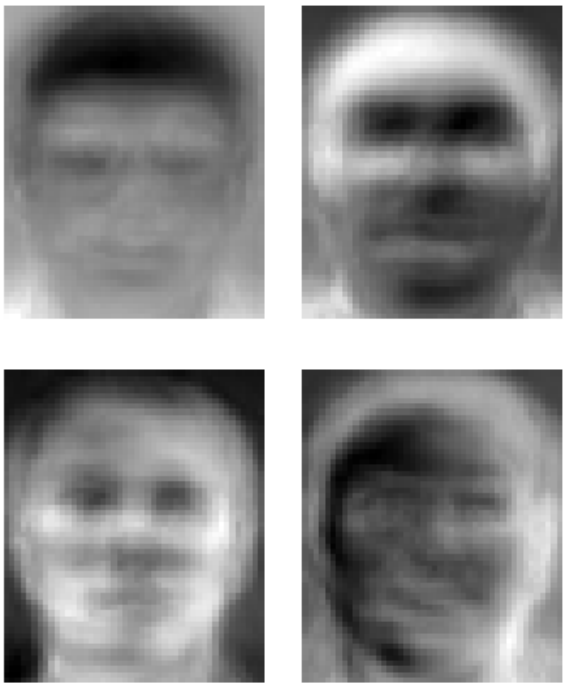


Figure 3.1: The first four eigenfaces of a set of face images, calculated from the Olivetti research labs face database

the face a human would identify as a feature, and indeed they do not in practise, as illustrated by Figure 3.1, but they are also clearly face-like in appearance. The basic premise with eigenfaces when attempting to identify a face image projecting it into face space is that performing a nearest neighbour search will find the best matching face from the database. This is assumed to be a good approach by many authors because of the observation (Meytlis and Sirovich, 2007) that images of the same face under similar conditions (e.g. lighting and pose) will be similar in terms of individual pixels and hence close to each other in image space. By using PCA to reduce the dimensionality of the space the nearest neighbour search is simplified and critically the (statistically) "important" characteristics of the images are still retained.

Many other authors have taken an approach to face recognition that uses the notion of image space as a theoretical basis for a sub-symbolic approach. One such work is that of (Bichsel and Pentland, 1994), where the authors present an

approach to what could be termed the *inverse image manifold problem*, i.e. given an image which is assumed to lie on a manifold can we determine the parameters that produced that image. In this paper a number of observations about the topology of the face image set are also made. The authors note that "Many transformations are approximately linear over small variations in the transformation parameters [...] which includes transformations such as small rotations, small scale changes in the illumination position or brightness". They also note that large scale transformations and translations typically result in discontinuities or high curvature. They point to (Turk and Pentland, 1991*a*) as evidence of these properties. Finally they use this to derive a method for face tracking, successfully run on a SPARCstation in real time.

In (Meytlis and Sirovich, 2007) the authors present their study into the dimensionality of face space, and recognition of faces in reduced spaces by human beings. Their conclusion is that for a "talented observer" roughly 100 dimensions are required, any less and too much of the detail required for recognition by a human being is lost. They also further note that an "average observer" requires $100-200$ dimensions of PCA to be retained in order that no quality loss is reported. This has implications for both human and computer recognition (whilst dimensionality reduction, e.g. in eigenfaces, is important it must be balanced against the need to retain enough information to make recognition possible still). There are further implications for attempts to model image manifolds themselves, particularly if the intended application is viewing by humans (any models constructed need to retain sufficient information to be discernible by humans). This result could be used as a guideline for both the human viewer and the recognition algorithm usages, however it is important to note that recognition by humans and algorithms are not the same and it may, at least theoretically be possible for an algorithm to outperform a human in this respect.

Despite the positive results reported by many of these authors there are a

number of problems when using eigen decomposition based vision algorithms in the real world. One of the major problems with such methods is the effect that changes in the background can have upon the recognition process. For example a front or profile view of a human face does not fill the whole of a rectangular image. Under controlled circumstances this is easy to deal with, by fixing the background, such as with a constant colour (which could be used to help control the capture process), or with depth of field of the camera. In less constrained scenarios (e.g. CCTV in a public place) the pose cannot be controlled, the illumination is still subject to variations (e.g. a hat may cast a shadow even if a light has been installed expressly to avoid shadows), the background is hard to control (e.g. passers by and other general changes) and the scale cannot always be controlled either.

Some of these problems can be addressed automatically, via a pre-processing step that attempts to normalise lighting, pose and remove the background from the image. However this is quite a hard problem to solve robustly. One such approach to solving some of these problems is presented in (Chang et al., 2007), where the authors propose a quadtree based structure to sub-divide images into a number of areas upon which eigen decompositions are performed. This has the effect of making algorithms more robust, by allowing matching to occur based upon a number of sub-images, some of which may not be occluded or filled with background clutter.

3.3.3 Manifold learning and dimensionality reduction

The manifold learning problem, where the parameters are not known but are to be estimated, is a significant area of research. The manifold learning problem involves attempting to "recover the low-dimensional nonlinear structure ... [from] datasets" (Tenenbaum, 1998). A large and ever growing volume of literature has been published on this topic. Here we discuss several of the more prominent works, and refer the reader to a number of survey papers (Robert and Richard,

2009; van der Maaten et al., 2009). The work of Tenenbaum et al. (2000) developed ISOMAP, which is a non-linear extension of classical multidimensional scaling (MDS). ISOMAP addresses the problems seen in MDS where the Euclidean distance between two high-dimensional datapoints can often be much lower than the distance on the manifold itself. The nonlinearity of ISOMAP is introduced by using graph shortest path algorithms to estimate *geodesic* distances as the sum of short (and locally linear) distances on the neighbourhood graph. Figure 3.2 is an example of the ISOMAP procedure being applied to a set of face images. As noted by van der Maaten et al. (2009), ISOMAP has a number of weaknesses. Firstly ISOMAP is not well suited to non-convex data , and secondly ISOMAP is prone to "short circuit" connections in the neighbourhood graph.

LLE (Roweis and Saul, 2000), which was proposed at the same time as ISOMAP, is somewhat similar. Where LLE differs from ISOMAP however is that instead of looking at the global neighbourhood graph it seeks to explain the manifold as a collection of small linear patches, which overlap and hence offer a global nature. To this end, where ISOMAP uses an estimate of geodesic distances LLE computes a weight matrix (often referred to as "reconstruction weights") by representing each of the inputs as a combination of its nearest neighbours. This approach is very local, and LLE seeks to preserve this local information in the low dimensional representation of the data. The local approach taken results in a much sparser weight matrix which can be computationally beneficial for larger datasets. Furthermore LLE does not explicitly make as strong assumptions as ISOMAP (which cause ISOMAP to handle non-convex scenarios poorly), although LLE still performs quite poorly in such circumstances. A variant of LLE, called Hessian LLE, has been proposed (Donoho and Grimes, 2003), which is explicitly designed to address non-convex data. However this is very computationally expensive and so rarely applied for large datasets.

Laplacian eigenmaps (Belkin and Niyogi, 2003) are again similar to LLE in that

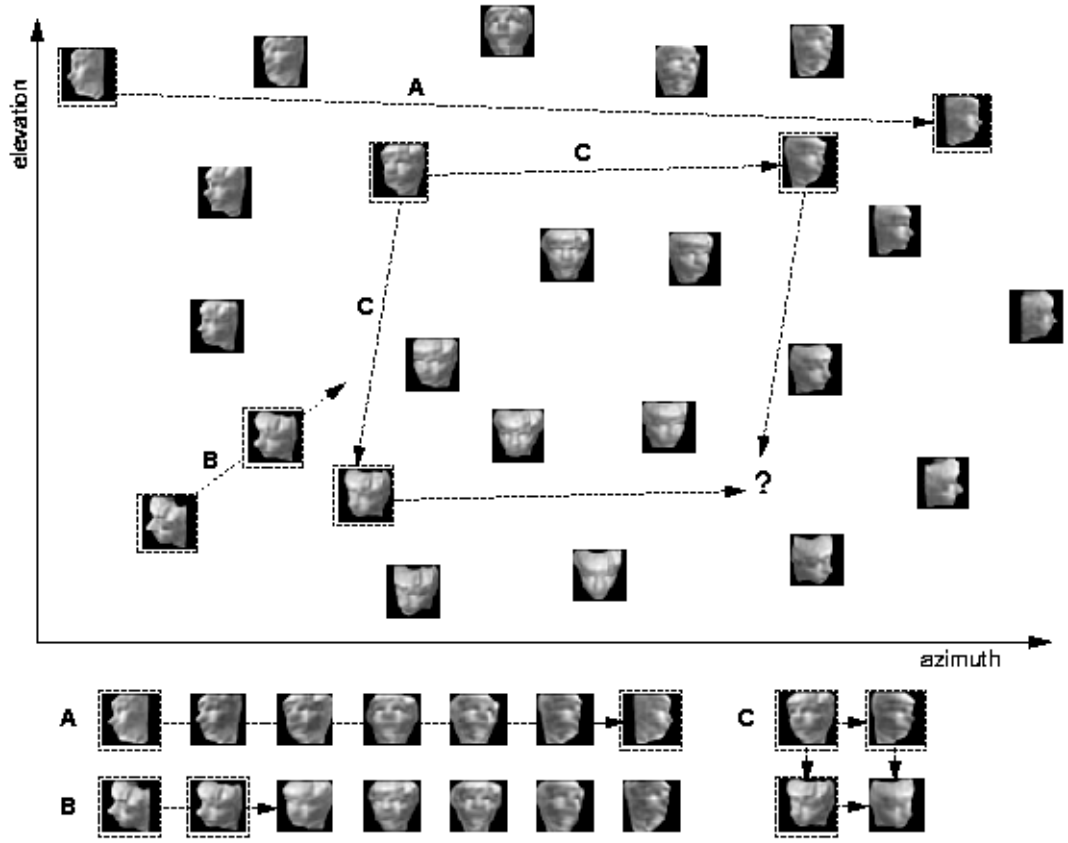


Figure 3.2: Mapping a manifold of perceptual observations, figure from (Tenenbaum, 1998)

the low dimensional representation is constructed to preserve the local properties of the manifold. The novelty of Laplacian eigenmaps comes from reformulating the objective function in terms of the Laplacian of the weighted neighbourhood graph. In this way spectral graph theory may be applied in order to efficiently solve this minimisation. Under certain circumstances LLE and Laplacian eigenmaps are theoretically equivalent, although this equivalence is rarely seen in practice.

For a number of applications once the manifold learning has been conducted as off-line processing step it is desirable to be able to identify the parameters of new images as they are seen. With PCA this is a trivial question of projection, however for algorithms such as LLE, ISOMAP, etc., this is not immediately possible. This important problem has however been addressed, in the general case (Bengio et al., 2004).

Manifold learning is often dependent upon a choice of image metric used. Souvenir and Pless (2007) look at the influence of the choice of metric upon ISOMAP, with a view to the specific problem of video datasets and consider several alternatives to the Euclidean distance.

Generative topographic mapping (GTM) (Bishop et al., 1996, 1998*b*) has been proposed as a principled alternative to self organising maps. GTM is also discussed extensively by Lee and Verleysen (2007), who state "In generative modelling, all variables in the problem are assigned a probability distribution to which the Bayesian machinery is applied". GTM aims to model the distribution in the data space (i.e. image space in this thesis) in terms of a smaller number of latent variables (the intrinsic parameters of the manifold). GTM introduces a discrete prior distribution in the latent space, typically a rectangular grid. Each node of this grid is chosen to be a Gaussian sphere to model the noise. This choice of a discrete prior simplifies the mapping between the data space and the latent space, which is formulated as an optimisation problem. This is initialised to be approximately equivalent to PCA, and solved with the use of an EM algorithm. The resulting model of the manifold ends up being a probabilistic (as opposed to a geometric) model in terms of a set of basis functions, which are Gaussians in the data space. The authors state that for a 2-D latent space they typically have $O(100)$ sample points within a small radius of the centre of each basis function. From the point of view of manifold learning the choice of the discrete prior, without regard for the underlying data, can be constraining for real-world applications. The probabilistic approach does however make it possible to detect problems in the resulting dimensionality reduction.

A number of authors (Bregler and Omohundro, 1994; Cho et al., 2003; Liu et al., 2006; Verbeek, 2006) recently have proposed synthesis of new images based upon manifold learning techniques. Verbeek (2006) proposes a probabilistic non-linear manifold learning method. Once learnt the proposed mapping offers a two-

way mapping between the (learnt) global parametrisation and the images themselves. The global mapping is obtained by "stitching" (combining) locally linear mappings using an EM algorithm, which seeks "agreement" (in a probabilistic sense) between the local models in the single, global, coordinate frame. The author notes that this is sensitive to the choice of parameter initialisation; poor choice of initial parameters can result in termination at local optima instead of the desired global one. The author uses LLE itself as an approach to selecting the initial parameters. The local linear models are learnt by approximating the noisy data using a number of components. There are a number of parameters which must be set in order to run this algorithm, including the latent dimensionality and the number of mixture components, although there are methods for automatically determining appropriate values for these. The resulting model is continuous in the latent (i.e. parameter) space, which means new views can be synthesised. The results reported from this compare favourably to corresponding results for alternative methods including GTM. Liu et al. (2006) propose a manifold learning based approach to the dynamic texture synthesis problem discussed in Section 3.2. Like Verbeek (2006) they propose learning the manifold as a set of globally coordinated, locally linear models. Their approach to learning the local, linear, models however is based upon PCA. One of the earliest papers to propose image synthesis as a manifold learning problem is introduced in (Bregler and Omohundro, 1994), where the authors propose an approach to manifold learning based upon "glueing" local, linear patches together. Here they use a clustering algorithm to estimate the centres of patches, and PCA to fit a patch to each of the clusters they discover. They introduce several methods for estimating short manifold trajectories on the locally linear patches, which corresponds to the synthesis of intermediate images. Another approach to face image synthesis is presented in (Cho et al., 2003), where the authors apply LLE to reduce the dimensionality of a set of face images. Once the dimensionality has been reduced they propose synthesis of new face images as

linear combinations of neighbouring images in the face feature space.

Using manifold learning to enable face image synthesis has been applied in other ways too. (Huang and Su, 2006; Su and Huang, 2005) apply manifold learning to synthesise new face expressions. Where this differs from previous work (Cho et al., 2003) is in the generation of new images; Su and Huang (2005) propagate optical flow between expressions, in a similar way to the way Souvenir et al. (2006) model the deformation in the image plane, as an alternative to the linear combinations of nearest neighbours proposed by Cho et al. (2003).

Several authors (Ham et al., 2006; Shon et al., 2005; Verbeek, 2006) have considered the problem of learning mappings *between* datasets, where two (or potentially more) datasets share the same low dimensional parametrisation, but the views themselves are different. Typically this is handled as an example of a larger class of manifold learning problems (e.g. Cevikalp et al. (2008); de Ridder et al. (2003); He et al. (2004); Wang et al. (2009)), which are referred to as "semi-supervised" or "supervised" learning. In general this problem is approached as an extension of standard manifold learning techniques, where two or more manifolds are learnt together. Shon et al. (2005) for example, propose a shared, latent variable space, which is learnt using an extension of a probabilistic manifold learning algorithm. This approach has been applied to a number of problems, for example robots imitating humans (Shon et al., 2005), finding mouth shapes that correspond to other people for visual speech synthesis (Ham et al., 2006), as well as general image retrieval problems (Cevikalp et al., 2008; de Ridder et al., 2003; He et al., 2004; Wang et al., 2009) where the aim is more generally to identify the class to which an input image belongs in order to find other, related, images. In general the techniques used to achieve this are extensions of standard techniques for dimensionality reduction; given that this thesis is concerned with dealing with individual manifolds these techniques offer us little insight into manifold modelling problem.

Aharon and Kimmel (2006) have approached speech synthesis and lip reading as a manifold learning problem. Here the authors propose using LLE and MDS to discover a low dimensional embedding of lip images. By considering the path taken in this low dimensional space by the articulation of different syllables parts of speech can be recognised, and furthermore the problems with simply concatenating these elements of speech (e.g. non-smooth transitions) can be avoided by analysing gaps in the low dimensional embedding. Using graph traversal algorithms in this space makes it possible to find and show existing, plausible lip images in a visually smooth way so as to avoid the problems of simple concatenation. Lip reading is performed as the opposite problem — given a sequence of images, which form one or more contours in the low dimensional embedding, find the best matching (previously learnt) contour(s) in order to identify the syllables and hence read the lips.

The manifold structure of datasets has been applied to a number of other vision problems, amongst them segmentation (Zhang and Pless, 2005; Zhang et al., 2006). Here the authors use standard manifold learning techniques (ISOMAP) to learn the parametrisation of MRI images. Once this has been discovered, like (Souvenir et al., 2006) they model, in the image plane, the deformations the heart in the images is undergoing. This model is used to facilitate segmentation, however the manifold learning applied is standard and the models of the deformation have already been discussed in the context of IBR, in Section 3.2.

From the perspective of this thesis manifold learning algorithms are less interesting than the vision algorithms — we already know the manifold structure and low dimensional mapping of all the datasets we will be considering, although again this work is just another piece of evidence for the image manifold concept.

Finally (Lu, 1998) is the most detailed study of image manifolds to date, which we will refer to more explicitly in a number of future contexts. Primarily though Lu (1998) makes a number of observations about image manifolds. Firstly a study

of the dimensionality of image manifolds is introduced, and concludes that the "ratio of pixel space [image space] dimension to image manifold dimension is of the order of 100 : 1". Another important result for our work introduced here is the conclusion that image resolution is a geometric invariant, i.e. the resolution of the images used does not alter the results observed (within reason). Finally the conclusion speculates that the concept of an image manifold may be inherently linked to the human visual system and human recognition of objects from novel views.

3.3.4 Conclusion

Many of the techniques discussed here are presented as face recognition problems. This does not however mean that they are limited to face recognition. Indeed some of the ideas such as ISOMAP have been applied to a wider range of machine learning problems including handwriting recognition (Tenenbaum et al., 2000), and even lip reading (Aharon and Kimmel, 2006). Additionally ISOMAP can be applied to cases where features have been extracted. Using eigen decomposition for vision problems has also been more widely applied than just face recognition, however face recognition is by far the widest studied, and representative of the other uses which is why we have discussed it here.

All of these vision methods are designed in some way to reduce the total amount of data, such that some direct matching can be reasonably performed. This would not be suitable for IBR techniques, where retaining as much information as possible is desirable, in order to synthesise the best possible novel views. However, the existence of a significant number of techniques that rely upon well-defined, predictable structures in image space can be viewed as at least anecdotal evidence for the idea of image manifolds. Clearly none of this work alone shows representing entire complex virtual environments via structures in image space as feasible, although some, such as the topological maps of robot environments, at

least suggest the feasibility of such an idea.

In the end though it could be argued, perhaps slightly perversely, that all appearance based approaches to vision problems are in-fact just extracting the smallest features detectable with the given sensor (camera), such that there is a 1:1 correspondence between pixels in images and features of objects.

3.4 Image manifolds studied

Here we now introduce the datasets we have collected for use through out this thesis. We will also briefly discuss some of their properties and show why they are important. A summary of the manifolds we use is shown in Table 3.1. Example images and a visualisation of the parameter space are shown in Appendix B. We define intrinsic dimension as the parameter dimensionality required to uniquely describe a sampled image. For instance the CIRCLE_3 manifold has an intrinsic dimension of 1 because only one parameter (angle) is required to describe the position, yet we recorded two parameters (x, y position on the floor of the lab) when the dataset was recorded.

The example image manifolds which we study throughout this thesis are designed to cover a wide spectrum of possible image manifolds. By including a wide selection of image manifolds, from a large number of different sources through out we aim to show that any conclusions we reach are as generally applicable as possible. If we had used just one or two examples for our studies it could be argued that our results are influenced by the choice of dataset more than the methods used for the experiments. In a number of instances however we are only able to use a subset of this data for a variety of reasons which we discuss in the appropriate places. Overall though we use a large number of datasets for most experiments, except for those which are regarded as proof of concept.

Image Manifolds					
Intrinsic Dimension	Parameter space dimensionality	Name	Number of images	Resolution	Source
4	4	GAUSS	160,000	50×50	
3	3	EXPT03	5575	200×200	(Fig. B.1)
	3	WINDOW3	21216	768×576	(Fig. B.2)
2	3	KNIGHT_FIGHTING	253	900×1200	(Fig. B.3)
	3	KNIGHT_KNEELING	253	512×512	(Fig. B.4)
	3	KNIGHT_STANDING	253	456×608	(Fig. B.5)
	2	2DTEAPOT	10000	1280×1024	(Fig. B.6)
	2	2DWOODBOX	10000	1024×768	(Fig. B.7)
	2	BMNOISE	400	20×20	(Fig. B.8)
	2	CHESSBOARD	2500	50×50	(Fig. B.9)
	2	FACES	400	92×112	(Fig. B.10)
	2	SINECOS	2500	2×1	(Fig. B.11)
1	2	CIRCLE_3	851	400×400	(Fig. B.12)
	2	CIRCLE_4	836	400×400	(Fig. B.13)
	2	CIRCLE_5	850	400×400	(Fig. B.14)
	2	IDRIS_CIRCLE	1685	200×200	(Fig. B.15)
	2	IDRISFIG_8	2156	200×200	(Fig. B.16)
	2	STRAIGHT_1	233	400×400	(Fig. B.17)
	2	STRAIGHT_2	237	400×400	(Fig. B.18)
	2	STRAIGHT_3	261	400×400	(Fig. B.19)
	2	STRAIGHT_4	261	400×400	(Fig. B.20)
	2	STRAIGHT_5	262	400×400	(Fig. B.21)
	2	STRAIGHT_6	261	400×400	(Fig. B.22)
	2	STRAIGHT_7	261	400×400	(Fig. B.23)
	2	STRAIGHT_8	261	400×400	(Fig. B.24)
	1	BRUSH	139	640×480	(Fig. B.25)
	1	IDRIS_STRAIGHT	820	200×200	(Fig. B.26)
	1	WOODBOX	2000	1024×768	(Fig. B.27)

Table 3.1: The image manifolds we study in this thesis

Our manifolds are diverse. We have manifolds which vary from one to four parameters. Of these manifolds a number are sampled from real sources. These include the robots navigating through a number of environments, the lab indoors (STRAIGHT_N, CIRCLE_N) and outdoors (IDRIS_N), a pan and title camera mount observing the scene from a window (WINDOW_3), in different directions, at a number of time intervals. Another real dataset was captured manually using an unmodified digital camera to take photographs of a paintbrush (BRUSH), which was rotated by approximately one degree between every image, over slightly more than half a rotation[2]. Additionally we have a number of ray-traced images where we expect the noise to be much lower than the real images, and the images are higher resolution and more densely sampled than was possible with real images. The ray-traced images include WOODBOX, where a wooden box is being rotated, 2DWOODBOX, where as well as rotating the box the camera is moved in closer towards the centre of the box as a second parameter and 2DTEAPOT, where one parameter is the rotation of the object, but the camera is also being translated for a second parameter, so effectively all the camera positions lie on a cylinder. One further dataset we use from the 'real' category is the KNIGHT_N series, which are from the Lightstage work[3]. In this data series images of a knight, in a number of different poses were captured with the lighting direction itself varying. The light direction is a point on a sphere, which can be parametrised using two angles. The samples from the image manifolds we study are all regularly distributed in parameter space, which makes later experimentation simpler.

Additionally we have a number of datasets designed to test the image manifold hypothesis itself, introducing scenarios where we know the changes from a small change in parameters will result in large, discontinuous moves in image space. These include CHESSBOARD, which is the two dimensional translation of a chess-

[2]This, intentionally, is clearly not periodic therefore, but this does not diminish its usefulness nor prevent us from using it in experiments later.

[3]From: http://gl.ict.usc.edu/Data/LightStage/ which stated "This data is provided "AS IS" for academic and non-commercial use."

board pattern. The hard edges of the chessboard contributing to large changes in image space, and introducing aliasing where the translations align. Further we have the BMNOISE dataset where the two-dimensional parameter is used to seed a random noise function to generate the images. In this instance we would expect there to be no discernible manifold structure present at all. Additionally the FACES dataset is based upon the Olivetti research laboratory faces database, which we have used in a slightly strange way: the two parameters that control the 'manifold' in this instance are the subject (i.e. person), which are arbitrarily ordered, and the pose of the person, which is again arbitrary in ordering. We intentionally consider the pose as a single dimension, even though here, in reality, pose is more than that for two reasons. Firstly we do not have more detailed pose information available with each image. Secondly, and most importantly, this does not fit the image manifold assumption which allows us to test and explore what happens when that assumption does not hold.

Finally we included two very artificial datasets, SINECOS and GAUSS. SINECOS is greyscale and has only two pixels, which are simply $\cos x$ and $\sin y$, where x and y are the two parameters that control the manifold. GAUSS is a Gaussian, in the image plane which is controlled by four parameters. These parameters are the coordinates of the centre x and y and the hue, h and saturation s of the colour in the HSV colour model, with $v = 1$ fixed. Due to the extremely large number of images this introduces this dataset will be treated separately from the other datasets during experimentation and discussed explicitly in Chapter 11.

There are a number of datasets (Verbeek, 2006) associated with other papers, which would have been interesting to use in order to compare results. Unfortunately our experimental procedures, Chapter 5 (see also page 142), assume that both a parametrisation and neighbourhood information are known. Whilst both of these can be learnt the neighbourhood information is potentially unreliable (consider the intersection point of IDRIS_FIG8 or any of CIRCLE_N). Furthermore

learnt parametrisation are likely to be non-uniformly spaced. This significantly complicates, with little benefit, constructing direct comparisons of the three methods we propose later. These methods are easier to fairly compare when applied to uniformly sampled datasets.

3.5 Conclusion

Whilst recognition and rendering problems are in some aspects related, the two problems are also different. Consequentially they require suitably different treatment for a number of aspects. For example, in the case of face recognition, small losses of detail such as individual strands of hair, can be very useful to allow for natural variation in hair between images. In the case of image based rendering however, such loss of details is not acceptable — the human visual system is complex and very capable of perceiving lost details, unless the losses are specifically controlled.

When using an image manifold in an application where being able to reconstruct images is important the highly non-linear nature of most image manifolds becomes increasingly important. Any assumption that the manifold is linear will, over all but the shortest of distances, result in poor reconstructions of images, with noticeable quality degradation.

Developing efficient in-memory representations of image manifolds has proven hard, and many researchers have used techniques like PCA to simplify the representations. This is acceptable if the intended use is recognition of images, however for the purpose of synthesising new images it proves to lose too many of the details that make images appear realistic to a human observer. A potential solution to this problem is to develop exisiting surface representation techniques to model these structures in image space directly, thereby representing all of the images that make up the manifold.

Throughout all of the IBR literature we have surveyed, which covers many dif-

ferent approaches to IBR from simple texture mapping or composition of sprites to state of the art modelling of dynamical systems, and mappings between image, one thing is clear: the proposed solutions impose a set of restrictions on the input images which are suitable and the methods used to generate them are based on domain specific modelling techniques. In the case of IBR techniques the assumptions made about the input images for example are free of occlusions (Gortler et al., 1996; Levoy and Hanrahan, 1996; Shum and He, 1999), location (i.e. either fixed or accurately calibrated positions) and/or path of the camera motion (Agarwala et al., 2005; Aliaga and Carlbom, 2001; Chen and Williams, 1993; Gortler et al., 1996; Levoy and Hanrahan, 1996; Shum and He, 1999). In yet more cases the types of changes between images is limited or assumed to be a deformation, e.g. (Souvenir et al., 2006). Often the goal is simply to synthesise *plausible* novel images or an extrapolation (Agarwala et al., 2005; Basharat and Shah, 2009; Doretto et al., 2003; Schödl et al., 2000; Zhou et al., 2009). Several techniques require the input images to be from unusual or specially modified hardware (Aliaga and Carlbom, 2001; Rademacher and Bishop, 1998) or provide depth-maps with intensity values (Rademacher and Bishop, 1998; Shade et al., 1998). In some instances the parametrisation of the input images is restricted in dimensionality. Almost all "classical" IBR techniques, which often present the illusion of more freedom than there really is, e.g. (Shum and He, 1999), impose such a restriction. In addition to classic IBR more recent work which focuses on time series inputs (Agarwala et al., 2005; Basharat and Shah, 2009; Doretto et al., 2003; Schödl et al., 2000) assumes this implicitly. Several of the works focusing on time series data recover higher intrinsic dimension models, however the parametrisation of these models is typically somewhat arbitrary and not directly related to the parametrisation of the underlying acquisition process in a way that would be understandable by a user. Likewise with manifold learning based techniques, Ham et al. (2006) note "... the resulting representations do not directly reflect the parameters of interest

such as pose parameters or joint angles", in their discussion of ISOMAP and LLE although by no means limited to just those examples (Belkin and Niyogi, 2003; Robert and Richard, 2009; Roweis and Saul, 2000; Tenenbaum et al., 2000; van der Maaten et al., 2009; Verbeek, 2006). This also presents a problem for using third-party datasets where images are supplied without a known parametrisation, e.g. (Verbeek, 2006), which would therefore have to be learnt prior to use. Some manifold learning techniques do not even automatically suggest a target dimensionality for the reduction process. We know this information explicitly and wish to build upon it. Many manifold learning techniques require estimates of neighbourhood sizes, or a neighbourhood radius (with our datasets we know the neighbourhood of the samples in parameter space, yet this neighbourhood might not be the same in image space and we wish to preserve and build upon what we know about the input images). The dimensionality reduction offered is often achieved by in some way discarding the least significant aspects of the input dataset, or conversely keeping the most likely parts (Bishop et al., 1998*b*; Szummer and Picard, 1996). Often authors seek to develop mappings between similarly parameterised manifolds (Ham et al., 2006; Shon et al., 2005; Verbeek, 2006), thereby enabling synthesis (or searching) of corresponding (novel) views on one manifold by examples from another, which still does not address the case of explicitly known parametrisations. This thesis aims to fill this gap we have identified, by proposing (Chapter 4), developing (Part III) and evaluating (Part IV) novel methods for representing image manifolds in order that new views may be synthesised. The goal of these models is to represent directly in image space both the speed and the position of the underlying manifolds themselves, where the parametrisation of the manifolds is known a priori. The methods we propose for this modelling are themselves logical developments of well-known, highly generic *geometric* (as opposed to, say, probabilistic) modelling techniques, which introduce no further assumptions about the data, beyond the image manifold assumption itself. These

methods offer higher order models, as opposed to the typically locally linear models used in recent work. By constructing our models directly in image space we avoid making any stronger assumptions about the data (e.g. distribution, underlying physical systems) and we are able to produce models which exactly capture the appearance of known sample points, without any loss, whilst interpolating the manifolds between these.

In this chapter we have introduced the concepts of image space and image manifolds. These two concepts are central to this thesis. We have shown examples of these concepts being successfully used. We will be building on what we have seen here, and the ideas we saw in Chapter 2. The two areas, of IBR and appearance based vision, can be linked together by the common thread of the plenoptic function. Using this idea as motivation, and the datasets we have introduced here, we develop new techniques for modelling image manifolds. The modelling of image manifolds amounts to modelling the plenoptic function, which clearly could be useful for both appearance based vision applications and IBR. We focus more on the general concept of modelling image manifolds, although given that these two applications are our motivators for this work we refer back to them from time to time. These two applications, which we have discussed here, but in particular the existing appearance based vision techniques, which already explicitly or implicitly assume the inputs to be image manifolds, provide a promising base upon which to build.

3.6 Summary

In this chapter we have seen:

- the concept of image space;
- the idea of an image manifold;
- some areas where these are useful concepts;
- how this fits with existing literature;
- the image manifold datasets we will use throughout this thesis.

In the next chapter we will:

- introduce the research question;
- set out a series of aims and objectives to address the question.

Chapter 4

Aims and Objectives

4.1 Introduction

To approach the problems discussed in Chapter 1 it is proposed than an object or environment be represented using a set of images that represent its appearance, instead of a geometric model. This is potentially advantegeous for a number of target application domains.

As already discussed in Chapter 3, images containing $w \times h$ pixels comprised of c colour components can be considered as points in the *image space*, an $w \times h \times c$ dimensional space. The appearance of an object is made of a set of images in image space that under certain conditions may encompass a low dimensional manifold embedded in image space. The dimensionality is fixed by the degrees of freedom of the observation of the object, typically position, orientation and illumination. The concept of *image manifold* was discussed in depth in Chapter 3.

The notion of using image space, which has grown out of vision work, as a solution to what is traditionally a graphics problem implies that this work is placed directly in the overlap between these two fields. In addition this work is further involved in the convergence of vision and graphics by the use of several classic graphical techniques such as NURBS and PDE surfaces to represent a path or manifold in image space rather than the geometry of a single object. This

concept can also be applied to certain computer vision problems.

4.2 The research question

The research question this thesis will answer is:

> "Is it possible to use extended variants of generic, geometric modelling techniques to construct *exact* vectorial representations of purely *the appearance* of scenes, objects and large environments without making any assumption about the images stronger than the image manifold assumption itself?"

Compared to state of the art IBR, this proposed technique is more generic (see Section 3.5 for a discussion of this). Our proposed methods are differentiated from recent advances in manifold learning, which generate two-way parametric models, e.g. (Bishop et al., 1998*b*; Verbeek, 2006), by a number of features. Firstly our proposed models are direct non-linear interpolations as opposed to the approximate, probabilistic models, which are often based upon learnt parametrisations. Secondly, our proposed models are designed to work with sparser samplings than other methods, in order to estimate probability distributions a larger number of samples are required. Bishop et al. (1998*b*) state they typically have of the order of 100 data points within a small distance of each basis function, Verbeek (2006) explicitly assumes that the sampling is dense enough that between samples the model is linear[1]. Furthermore, unlike (Bishop et al., 1998*b*) we consider situations where we deliberately choose not to use a uniform sampling grid in order that the samples may be more intelligently selected to reflect the underlying data.

In order to address the research question it has been broken down into a series of smaller aims and associated objectives outlined below.

[1]We consider piecewise linear interpolation between sample points as the starting point for our experimentation in Chapter 8. This is then subsequently used as a reference point for comparison when we introduce higher order models of the manifolds.

4.3 Aims and objectives

The aims of this work are outlined below:

1. Investigate the properties and theory of 'image manifolds'.

2. Investigate possible generic methods of exactly representing manifolds embedded within image space.

The aims listed above have been further broken down into a series of objectives, detailed in the remainder of this section.

4.3.1 The properties of image manifolds

"Investigate the properties and theory of image manifolds"

1. **Demonstrate feasibility of geometrically representing image manifolds.** (Chapters 6, 7, 11)

   To date there is little truly conclusive work showing the feasibility of such an idea — most existing studies focus on simple, single cases. Throughout this work we consider a wide range of datasets, some which we claim are good examples of image manifolds, others where this is not the case. This allows us to experimentally verify our approach to modelling image manifolds and feed this back into our notion of what constitutes an image manifold.

2. **Investigate implicit assumptions in image manifolds.** (Chapter 11)

   The notion of an image manifold contains several implicit assumptions. These assumptions are important because in some cases they will not hold. Being able to explicitly identify these assumptions, and therefore consequentially also the cases where they do not hold would be very useful.

3. **Investigate visualisations of image manifolds.** (Chapter 7)

   Working with any high dimensional dataset it is useful to be able to visualise

in some way the data and any experimental results. It is therefore important that we develop and implement a suitable visualisation technique at an early stage in this thesis in order that the results obtained may be better understood.

4.3.2 Representing image manifolds

"Investigate possible methods of representing manifolds embedded within image space"

1. **Extend surface modelling techniques** (Chapters 9, 10 and 12)

   There are a large number of known generic surface modelling techniques, however in order to apply any of these to the problem of modelling image manifolds in image space we will first have to develop extensions to these in order to increase both the intrinsic and extrinsic dimensionality of the surfaces and the space in which they are modelled, as compared to typical surface modelling problems.

2. **Implement the surface representation techniques.** (Chapters 9, 10)

   Implementations of the various image manifold modelling techniques are required to enable further experimentation with the techniques. Implementations need to be reasonably efficient and numerically robust to enable experiments to progress at a reasonable rate.

3. **Investigate the implemented surface representation techniques.** (Chapters 9, 10)

   All of the surface representation techniques will require decisions to be made at many points during their implementation and usage. There are a myriad of (usually) domain specific methods for this in the literature, which address issues such as 'are we interpolating or approximating our sample points?' and 'is this global or local?'. Choices like these and other less generic, technique

specific ones need to be made. All of these decisions will impact on the representation of the image manifold, and its corresponding visual fidelity. It makes sense therefore to investigate some of these questions further with a view to the specific application.

4. **Perform an extensive evaluation of the implementations.** (Chapters 5, 11)

   Constructing exact models of surfaces in high-dimensional spaces, with known parametrisations, is an important area. In particular there is little existing work looking at the general case of *geometric* modelling of image manifolds in the full image space and not some reduced subspace. Evaluation, therefore must be comprehensive and should be conducted with reference to both visual and numerical accuracy. Visual and numerical accuracy are not the same, and given that at least one possible use of image manifold models is producing images for humans, the human visual system itself is potentially equally important to numerical accuracy. We seek a principled and objective measure of the successes of our proposed models, and therefore propose the use of image metrics to address this.

5. **Investigate strategies for discarding sampled points from an image manifold**. (Chapters 6 and 12)

   Throughout the computational elements of this work we shall be using discretely sampled image manifolds. It is well known that insufficient samples can lead to aliasing problems when reconstructing from these samples at a later point. The Nyquist-Shannon sampling theorem states that at least $2 \times f_{\max}$ samples are required to accurately reconstruct the signal, where $f_{\max}$ is the highest frequency of the signal. We will assume (out of necessity) that we have more than enough samples available to at least provide a good approximation of the image manifold itself. In our case we do not know $f_{\max}$, we (usually) do not have infinite signals and our goal is not always to

produce an exact representation of the image manifold itself. The question then is how to select the best (i.e. small whilst still preserving most of the signal) subset of our available samples to use in our experiments.

4.4 Summary

In this chapter we have presented:

- our research question;
- our aims, which will address the research question;
- a further series of objectives that will enable us to complete our aims;
- an outline of how the chapters of this thesis correlate with individual objectives.

In the next chapter we will explore:

- some general properties of image manifolds;
- how these properties might influence the remainder of our work;
- some of these properties for our specific example image manifolds.

Chapter 5

Methodology

In Chapter 3 we have seen a large selection of datasets, which we propose can all be treated as image manifolds. These image manifolds cover a diverse range of sources and a correspondingly diverse range of properties as a result.

In short we have a rich and varied selection of data on which to perform experiments to address our research question, as well as more theoretical avenues. This breadth and variety allows us to isolate our findings from the choice of datasets used to conduct the experiments to a large extent.

The problem which then presents itself, is how to set about constructing a series of experiments to answer the research question. The theory and topology of the set of all natural images (Carlsson et al., 2008), whilst interesting by itself, is likely to prove a dead end for answering the question. This is in part due to the fact that an image manifold is a greatly reduced subset of the set of all natural images and so any conclusions we may draw from work on the set of all natural images will be limited in scope and consequently not incorporate the knowledge that we are interested in a reduced subset. Another potential approach, the one taken by this thesis, is to actually construct a variety of different vectorial models of image manifolds, for which as we shall see later on is non-trivial. As well as this, with every vectorial model we construct there will be inherent choices to be made regarding the construction of the model, choices which will inevitably influence

the fidelity of the resultant model. Once we have constructed a model, as we will be doing throughout most of this thesis, we then require a principled approach to evaluating it. That is to say, we wish to be able to assess the quality of the model and try to show that it is indeed a better (in either a numerical or visual sense) vectorial model of the image manifold, as we hoped.

In all cases, except the case where we use all the samples from the image manifold, there will be a number of sample images which were not used in the construction of the vectorial model of the image manifold, but which we know do lie on the manifold. Furthermore we will also know the parameters of these samples. We can request that our models generate values for these unused images, which provided the model is faithful to the underlying manifold, should correspond to the known samples that we have in our dataset. In this way, by constructing our vectorial models of the manifolds using a subset of the samples available to us, we can use the remaining samples as a ground truth with which to perform some evaluation.

This in turn leads us to a problem though: how should we evaluate the images? Our evaluation must be robust and repeatable, but furthermore since our entire motivation for asking this question is detached from any single specific application it must be generic, yet representative of a wide range of possible applications.

5.1 Image metrics

The question of a principled approach to evaluating a vectorial model is therefore a central problem for this thesis. We find this problem to be further confounded by the simple quantity of images in question. Assuming we have 3 different models, each of which have 5 different parameters, which we wish to try with 5 different values for every single dataset that produces

$$3 \times 5 \times 5 \times 21 = 1575, \tag{5.1}$$

models to construct. Furthermore if the average dataset has $10,000$ images then the number of evaluations we have to perform will be of the order of 10^7!

This effectively rules out immediately any full scale evaluation by human observers. Humans are notoriously bad at objectively, repeatably and quantitatively evaluating images at the best of times. Given the sort of numbers in question devoting just 1 second to evaluate each of the anticipated output images would take more than 115 days of non-stop work to evaluate them all! Even if that were possible producing any results which were demonstrably significant would still further complicate things and multiple[1] evaluations would be required.

In order to effectively evaluate large quantities of images we therefore turn to metrics. In metrics we find the principled, objective and repeatable evaluation method we seek. Furthermore metrics provide an interesting link up with the theoretical side of metric spaces, topology and manifolds.

5.1.1 Fundamentals

The concept of a metric dates back almost as far as numbers themselves and while almost certainly not originally seen as such the invention of subtraction produced the first, most simple metric. Consider, as a trivial example, the number line of natural numbers, illustrated in Figure 5.1.

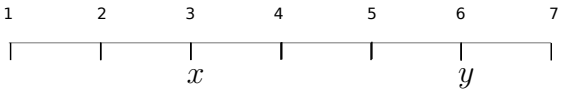


Figure 5.1: The numberline of natural numbers.

If we want to answer the question "How far apart are the two numbers y and x?" then the answer is simple, we just use subtraction $y - x = 6 - 3 = 3$. The answer is three, because we have to move 3 units further down the line to go from one to the other. When we try to answer the opposite question, it is obvious what

[1]Employing one single person, at minimum wage would cost approximately 16K! I'm told unions object to not giving workers breaks as well!

the answer should be, but simply applying $x-y = 3-6 = -3$, so we must consider the absolute value, $|x-y|$ to give us the answer we expected. Clearly the distance between any given number and itself must be 0. This illustrates simply the first three axioms of a metric, D:

- $D(x, y) \geq 0$ (*non-negativity*)
- $D(x, y) = D(y, x)$ (*symmetry, commutativity*)
- $D(x, y) = 0 \iff x = y$ (*identity, indiscernibles*)

What happens when we want to apply this in 2-space? It is easy to show, by application of Pythagoras' theorem that the distance between any given pair (x, y) of points in a plane is given by

$$D(x, y) \quad = \quad \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}. \tag{5.2}$$

This is the Euclidean distance in 2-D. This leads us to the fourth[2] axiom of metrics, namely if we have 3 points, x, y, and z:

- $D(x, y) + D(y, z) \geq D(x, z)$ (*triangle inequality*).

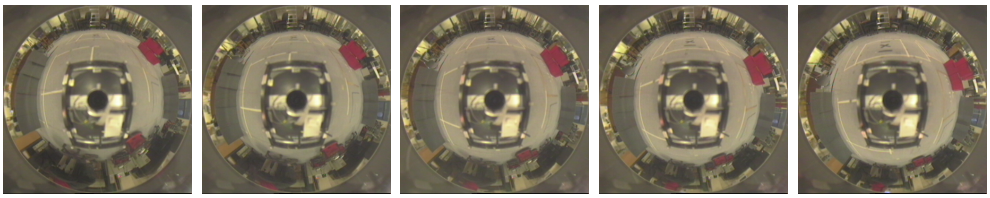
We can see this to be true, by considering the case of three points in a plane. If we wish to minimise $D(x, y)$ and $D(y, z)$ without altering x and z the only option is to move y. The only way we can make it shorter is by moving it towards the line between x and z, until it lies upon that line.

There are a good number of texts on Metric Spaces, e.g. (Copson, 1968; Reisel, 1982; Victor, 1985), which discuss this definition of a metric much further.

5.1.2 Intrinsic metrics

We all intuitively think of the universe as being 3-D in nature. When someone asks a question like "How far is it to travel between the two poles of the earth?" the

[2]There is actually some redundancy within these axioms and only three are really needed to derive the rest.



(a) (b) (c) (d) (e)

Figure 5.2: Counter example (from STRAIGHT_5 dataset) for Euclidean distance as intrinsic metric for image manifolds

answer seems simple at first. The distance between the two points, as calculated by the Euclidean distance assumes it would be possible to travel through the centre of the Earth, and it is not suitable to answer the question like this. This leads us to another important concept, *intrinsic* metrics. Really in this case a more useful metric would be one that factors in the journey around the circumference of the Earth. This also leads us back to a key concept we introduced previously, in Chapter 3. From the perspective of a human walking on the Earth it seems flat, locally Euclidean geometry works on the surface. This is because the Earth is a 2-manifold, embedded in 3-space. That is to say the Earth is a good example of something which is a metric space (and hence also a topological space), which is locally Euclidean.

What would an intrinsic metric look like for an image manifold? If we knew the complete path in image space that (for example) a translation or rotation of a camera produces we could approximate an intrinsic distance as simply the sum of the Euclidean distances. Figure 5.2 proves, by counter example, that the Euclidean distance metric is not in general an intrinsic metric on an image manifold. If it were the case that the Euclidean distance metric was an intrinsic metric in this case we would expect to see $D(a, e) = D(a, b) + D(b, c) + D(c, d) + D(d, e)$ because by definition a path between a pair of images could be made by using successive images on the manifold.

Proof.

$$D(a,e) \stackrel{?}{=} D(a,b) + D(b,c) + D(c,d) + D(d,e) \tag{5.3}$$

$$19068.5 \stackrel{?}{=} 11200.6 + 11261.1 + 10753.1 + 11970.5 \tag{5.4}$$

$$19068.5 \neq 45185.3 \quad \square \tag{5.5}$$

5.1.3 Evaluating metrics

Given that we propose to use image metrics to evaluate our synthesised images, it is only right that we broaden our search for a metric beyond the usual L2-norm (Euclidean distance). We must consider the issue of what makes a good metric. The following criteria are therefore proposed as candidates for metric evaluations:

1. How well does it approximate an intrinsic metric on the manifold?
2. How well does it fit with our intuitions as human observers?
3. How well does it fit with the application in question?

For the first criterion we can construct scenarios which test the metric. Given the fixed, limited number of samples (q) available to us the best estimate of an intrinsic metric we can construct is the sum of the distances between every consecutive pair of points on the shortest path,

$$D_{\text{INTRIN}}(q_n, q_m) = \sum_{i=n}^{m-1} D(q_i, q_{i+1}). \tag{5.6}$$

Ideally we would select a metric $D(q_n, q_m)$, where $D(q_n, q_m) = D_{\text{INTRIN}}(q_n, q_m)$ $\forall n, m \in q$. In practice however it is unlikely that we would be able to find such a metric. It may however still be interesting to compare the relative differences between the best estimate intrinsic distance and the individual distances reported (i.e. $D_{\text{INTRIN}}(q_n, q_m) - D(q_n, q_m)$. It would not be possible or reasonable to perform this test for every possible pair of images for every one of our datasets, however we

will perform this for a small number of samples and consider it in our evaluation of the metrics we propose to use.

The second criterion is obviously somewhat harder to quantify. We can, and do, construct a number of scenarios where the indications from the Euclidean metric clearly do not correlate with the natural expectation for an image metric. The hope being that one or more of the alternative metrics we consider will offer an improvement in this area over the Euclidean metric.

Finally, with regards to the third criterion we have already discussed earlier in Chapters 2 and 3 a number of applications for image metrics. This included the visual compass (Labrosse, 2006) where the author proposed measuring distances between images as an alternative to a traditional magnetic compass. We will not discuss the method itself much further here, since we have previously covered it, however it is useful as a sample (vision) application of image metrics from the point of view of our criteria. In this instance none of the metrics we will consider are likely to offer a strong candidate for replacing the Euclidean metric, since they are all significantly slower, which is unhelpful for an application in robotics. After Section 5.1.4, in which we outline a number of possible metrics, we present in Section 5.1.5, a discussion of some of the proposed metrics in the light of this example application.

5.1.4 Alternative candidate metrics

Metrics have found uses in a great number of fields and problems, for instance Li et al. (2003) propose a normalised information distance as a metric for genome comparison, as well as automated language tree computation. In this thesis we will be focusing on the metrics more relevant to the fields of graphics, vision and visualisation, which has again seen a diverse range of applications. In (Zhou et al., 2002), for example, the authors propose a number of metrics for evaluation of the results of visualisations of a rheology experiment. They consider a number

of different metrics, which they classify as operating in the spatial domain, the frequency domain or as a perceptual metric.

Here we choose to consider all of the metrics we are interested in henceforth as belonging to one of four categories, namely Classical, Statistical, Perceptual and finally Retrieval. The latter category, however they are generally not suited to our problem and we do not discuss it further. Throughout the remainder of this section we present a brief overview and discussion of some more common metrics for each of the retained three classes. Note that in several cases we relax the axioms we have previously discussed in one way or another. This means that in the strictest sense of the term many of these are not really metrics. This will be noted as appropriate.

5.1.4.1 Classical

This first category of metrics could also be called the "geometrically derived" metrics. The three most common metrics in this class are the Euclidean distance, the Manhattan distance and the Chessboard distance. These are commonly also referred to as L-norms.

We have already seen the Euclidean distance in 2-space, the length of a line in a Euclidean space, between two points x and y in Equation 5.2. The Euclidean distance metric, which is also commonly referred to as the $L2$-norm is trivial to generalise, with x_i and y_i being the individual components of x and y respectively, and d being the total number of dimensions for x and y (which must be compatible):

$$D(\underline{x}, \underline{y}) = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2}. \tag{5.7}$$

In many instances, where only the ordering is important and the magnitude of the distance itself is not important the square-root operation may be omitted.

As this can be an expensive operation it is often done, e.g. in real-time robotics applications.

Manhattan distance, also called taxicab distance or L1-norm, is very similar to the above. It can be defined as the number of city blocks in an idealised, grid-based city required to move between two given points:

$$D(\underline{x}, \underline{y}) = \sum_{i=1}^{d} |x_i - y_i|. \tag{5.8}$$

This has again found favour with real-time applications as an approximation of the Euclidean metric, e.g. (Mitchel and Labrosse, 2004), due to the very small implementation costs.

Finally, and closely related to the Manhattan distance, the Chessboard distance is defined as the number of moves a King would have to make on a chessboard to get between two points:

$$D(\underline{x}, \underline{y}) = \max(|x_1 - y_1|, |x_2 - y_2|, \ldots |x_{d-1} - y_{d-1}|, |x_d - y_d|). \tag{5.9}$$

One simple advantage these last two metrics offer over the more conventional Euclidean metric is computational cost — there are only simple addition, subtraction and comparison operations[3] required to implement them which makes it more suited for environments where processing resources are scarce. In addition the distances calculated can be comparable to the Euclidean metric. Examples of the use of this class of metrics includes (Bozkaya and Ozsoyoglu, 1999; Kim et al., 2001; Labrosse, 2007; Neal and Labrosse, 2004).

5.1.4.2 Statistical

In this section we consider a number of candidate metrics, which arise from the fields of statistics and information theory. Many of these metrics have been used to

[3]The relative costs of each will vary from platform to platform.

evaluate compression algorithms, because of the obvious connections. For instance mean square error and peak signal to noise ratio are often used as metrics for compression.

Mutual Information is widely used e.g. (Chen and Varshney, 2003; Maes et al., 1997; Russakoff et al., 2004), as a metric when performing registration of images, or data of different modalities. The entropy of a discrete random variable, A (which could be an image) is given by

$$H(A) = -\sum_{a \in \mathbb{A}} p(a) log_2 p(a), \tag{5.10}$$

where $p(a)$ is the probability of A being in state a with a given alphabet $\mathbb{A}$. In the case of a digital image, with some given colour depth d (bits per pixel) there will be 2^d possible values for any given pixel. This means that the theoretical maximum entropy reported will be $log_2 2^d = d)$ bits, when all colours are equally possible.

The joint entropy of a second, unrelated and hence completely independent discrete random variable B would be given by

$$H(A, B) = H(A) + H(B). \tag{5.11}$$

In the general case, when the random variables are not independent the *joint entropy* is given by

$$H(A, B) = -\sum_{a \in \mathbb{A}} \sum_{b \in \mathbb{B}} p(a, b) log_2 p(a, b), \tag{5.12}$$

where $p(a, b)$ is the joint probability and $\mathbb{B}$ is the alphabet of B, which for two 24-bit colour images would be the same in both. The *mutual information*, how much information is shared between the two random variables, is therefore given

by

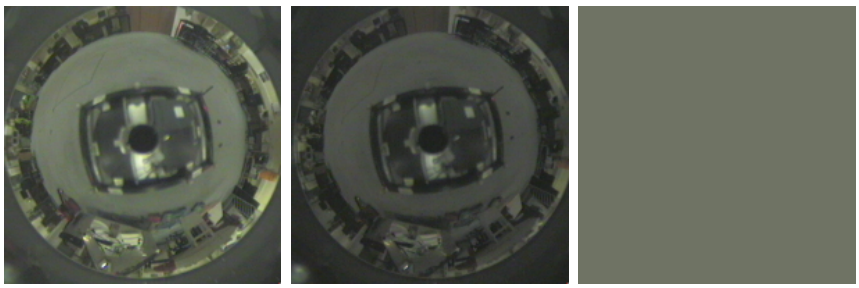
$$MI(A,B) \quad = \quad H(A) + H(B) - H(A,B). \tag{5.13}$$

In the case of two independent variables $H(A) + H(B) = H(A,B)$, hence the mutual information is zero.

For experiments performed henceforth a robust and efficient implementation was sought. For this reason the widely used implementation provided in the widely used "Insight Segmentation and Registration Toolkit" (ITK)[4] is applied.

Levenstein distance (Levenshtein, 1966), also referred to as the edit distance, counts the minimum number of operations required to transform one string into another. Usually the possible operations consist of insertion, deletion and replacement. In terms of time complexity this is the most costly of all the metrics discussed here.

5.1.4.3 Perceptual



(a) Image A $(0,0,1)$ on Expt03 (b) Image B $(0,0,5)$ on Expt03 (c) Image C

Figure 5.3: An case where the behaviour of the Euclidean distance metric is arguably not what would be expected perceptually

Until this point we have not considered an important aspect in many applications of image metrics — the human visual system. In (Girod, 1993) the authors note that very often, in compression, elaborate statistical models of sources of images and the images themselves are produced. Yet crude visual metrics are still

[4] www.itk.org

used. An example of one such simple disparity is illustrated in Figure 5.3, where, according to the Euclidean metric at least, image A is more similar to image C than to image B. They further observe that this is in fact, a very poor model of the human visual system and there are significant disparities between what human observers perceive and what traditional measures such as mean squared error report. The authors further argue that in the case of image coding (of which image manifold modelling might be thought of to be an example) the known perceptual effects should be considered. Metrics in this perceptual class attempt to address the disparity between the human visual system and traditional metrics by building upon the rich, but by no means complete, body of literature which exists on the human visual system itself.

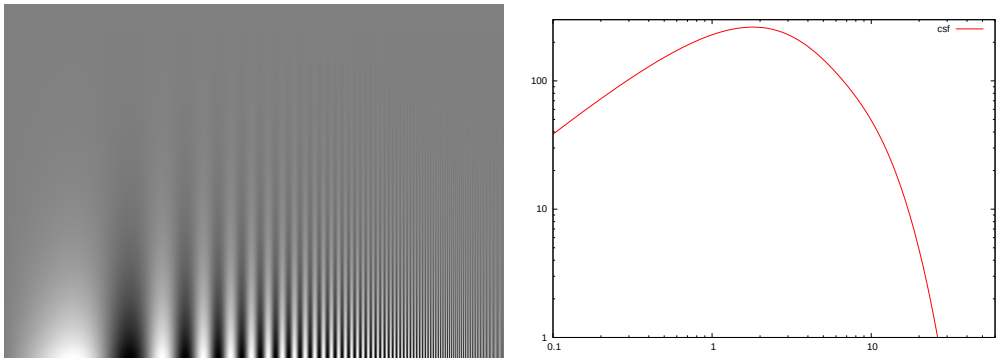
Although a complete end to end model of the human visual system does not exist, many individual phenomenon are very well studied and modelled. This knowledge can be put too good use in this context. It is interesting to note here that the human visual system does not make a metric in the strictest sense. There are many examples of images which can be constructed that look indistinguishable to an observer, yet are clearly not identical. This could also lead to instances where the triangle inequality is not satisfied, for instance an image y could be constructed that is very similar to, or indistinguishable from, another two images x and z, yet x and z both clearly being distinct images.

In the general case perceptual metrics attempt to identify and disregard differences that, although numerically noticeable, would not be noticed by an observer.

Here we focus on a number of these features, as used by the authors of (Yee and Newman, 2004), which is based upon (Daly, 1993). Firstly it is widely known that the RGB colourspace is not perceptually linear. The CIE $L^*a^*b^*$ colour model is a perceptually uniform colourspace, which is therefore used for the remainder of these models.

The contrast sensitivity function (Figure 5.4) and masking function is one

example of a well studied and documented feature of the human visual system. Depending upon printing, but primarily viewing conditions Figure 5.4(a) will look different. Sensitivity to changes in contrast by humans is a function of the frequency, the viewer's age and the viewing distance itself. Figure 5.4(b) shows a plot of contrast sensitivity at differing frequencies.



(a) Contrast Sensitivity Chart (Campbell and Robson, 1968)

(b) Contrast Sensitivity Function (Barten, 1990)

Figure 5.4: Illustrations of a well understood perceptual phenomenon, as used in (Yee and Newman, 2004)

There are many other features which have been identified, discussed and modelled in the psychophysical literature. For example the effects of motion, e.g. in video sequences, has been well documented and potentially could influence an observer of the images we produce in this work. We make the assumption here however that images are static, stand alone entities. This allows us to further discount other perceptual effects such as chromatic adaption.

An alternative approach to metrics, which focuses less on the models of the human visual system itself, but more on the features contained within the images could also be used here. We propose a method of comparing two images, by which a feature detection algorithm of some description is run on both images. The motivation for this approach is simple: if a human were asked to devise a repeatable method of quantifying the difference between a pair of images, selecting features and comparing the positions of the features would be a sensible approach. Furthermore given a suitably chosen feature detector and two relatively similar

Table 5.1: Summary of metrics

Euclidean	Equation 5.7
Taxi	Equation 5.8
MI	Equation 5.13
Pdiff	(Yee and Newman, 2004)
SIFT	Algorithm 5.1

images one would expect the features found in both to be broadly similar. Given a deterministic feature detection algorithm the features found in two identical images would be identical.

Having found a set of features in both images a comparison, using the Euclidean distance, is subsequently performed between the sets of features returned. This procedure is given in Algorithm 5.1. Potentially the images we wish to compare will have undergone scaling or rotations. With this in mind we have opted to use the Scale Invariant Feature Transform (SIFT) of (Lowe, 1999). Our results, presented in the next section indicate that this does achieve a comparable performance to the other perceptual metrics.

Algorithm 5.1 SIFT as an image metric

```
d ← 0 {Total distance is initialised to 0}
for p_a ∈ A do
  t ← inf
  for p_b ∈ B do
    if D(p_a, p_b) < t then
      t ← D(p_a, p_b) {When the distance between this pair of points is the best
      so far consider that instead}
    end if
  end for
  d ← d + t {Add the distance of the minimal pairing to the total distance}
end for
```

5.1.5 Evaluation of the metrics

In this chapter we do not attempt to derive any single rule or threshold for any of the given metrics at which we can say that quality has in some way degraded beyond an acceptable level, neither do we attempt to derive the opposite. Instead

we leave the evaluation more open — after all for some applications (e.g. vision) it may be sensible to allow a much higher error to exist than in the case of an immersive virtual environment.

To put our selected metrics into a more real context the following tests have been run:

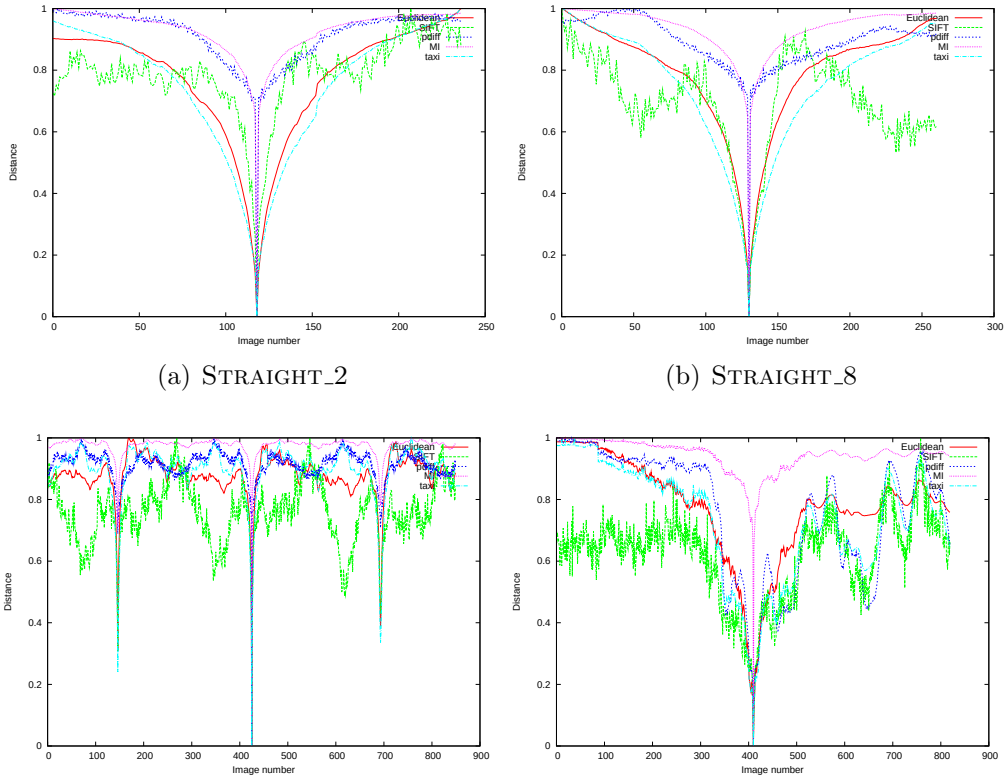
1. Visual navigation function.
2. 'Intrinsicness' test. (See Figure 5.2)
3. Plain colour image comparison. (See Figure 5.3)
4. Noise introduction

5.1.5.1 Visual navigation with other metrics

We have constructed a test intended to be similar to the visual navigation. In doing this we take all of our 1-D image manifolds, and take the central images, with the central image being half way through the series of images. In the case of a robot moving a 10 meter path in a straight line at a fixed speed and capture rate the central image would occur 5 meters through the path. We then compute distances for every image along the length of the manifold compared against the central image. Normalised plots of this are presented in full in Appendix C. Presented and discussed in Figure 5.5 are a selection of these results. We refer to the mutual information measure as *MI*, and the perceptual metric as *pdiff*.

Results from the Levenstein metric have been omitted here on the grounds of speed — it is many times slower than all of the other metrics combined!

We notice here that the first two (Figure 5.5(a) and Figure 5.5(b)) look as expected — in each case for all of the metrics we observe a approximately symmetric v-shaped valley function which drops off sharply around the central image. This exactly hits 0 for all metrics, which confirms firstly that $D(x, x) = 0$ as we would expect. In all the cases, both here and in the appendix the Euclidean and taxi



(a) Straight_2 (b) Straight_8

(c) Circle_5 (d) Idris_straight

Figure 5.5: Visual navigation metric experiment

metrics seem to be almost interchangeable. This confirms what was reported in (Mitchel and Labrosse, 2004). The SIFT metric seems to be the least smooth of all metrics, likely explained by the fact that we are merely seeking a mapping between the two sets of features returned, which will vary both in quantity and position between even neighbouring images. In both of the first two cases the perceptual metric and MI closely follow each other. The Circle_5 results are interesting — we know from the parameter space data (the robot position) that the robot performed three complete circles, although they were not completely identical. We can see this in the results, from all of our metrics, represented as the two minima either side of the centre. Idris_straight also performed as expected, especially given that it was captured in a real, live dynamic environment, with people and cars moving in the background explaining the majority of fluctuations observed.

Clearly if this were intended as a real-time vision algorithm, with some sort of

Table 5.2: 'Intrinsicness' results

Metric	$D(a,e)$	Σ	$D(a,b)$	$D(b,c)$	$D(c,d)$	$D(d,e)$
Euclidean	19068.5	45185.3	11200.6	11261.1	10753.1	11970.5
Pdiff	125017	457476	110663	115740	113287	117786
SIFT	3578.46	11050.88	2798.02	2817.52	2318.76	3116.58
Taxi ($\times 10^6$)	8.46	18.43	4.71	4.67	4.33	4.72
MI	0.873	3.093	0.779	0.780	0.763	0.771

Table 5.3: Plain colour comparison results

Metric	$D(A,B)$	$D(A,C)$
Taxi	3.79×10^6	3.28×10^6
Euclidean	12698.9	11406.9
Levenstein	116274	117149
SIFT	1725.1	∞
MI	0.8835	0.9755
Pdiff	19819	34762

gradient decent function searching for a minima at an unknown location the wider and smoother the valley (and hence shallower and more predictable the slopes on the sides) the easier it would be to find the minima. All of the metrics which we have introduced here offer little performance improvement and in most cases are notably steeper around the minima than Euclidean or taxi. This suggests neither offer an algorithmic improvement, nor a speed improvement. It is still interesting to note that all of the metrics offer very similar shape functions here.

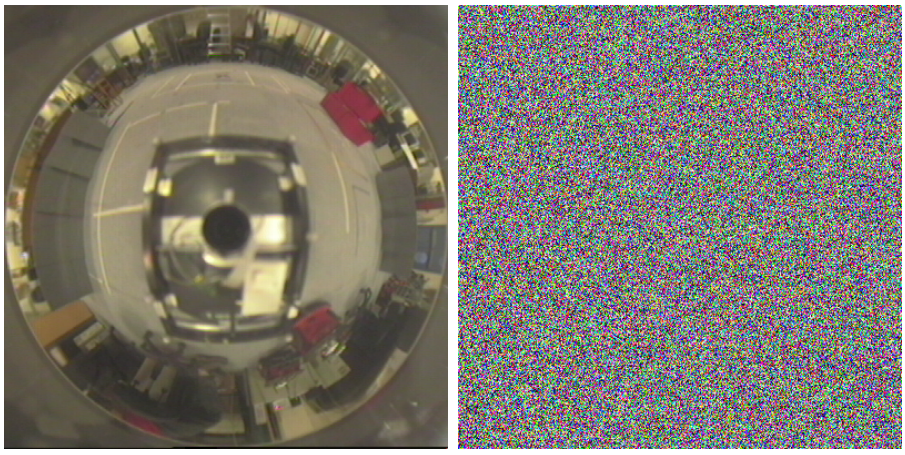
5.1.5.2 'Intrinsicness'

Table 5.2 shows the results from repeating what we did in Figure 5.2 with the rest of the metrics we have proposed to use for this thesis. In all cases the sum of the distances is greater than double the distance direct from a to e, whilst in several cases (MI, pdiff) it is closer to four times the distance. Clearly therefore none of these metrics is close to intrinsic on the manifold in questions.

5.1.5.3 Plain colour comparison

Table 5.3 shows the results of our metrics on the images in Figure 5.3. Notice here that the SIFT metric failed, owing to the fact that in a completely smooth image there are no features to be found at all. It is reassuring to note that the metric which claims to be perceptual returns a distance between the two images from the robot which is lower than the distance between the first image and the plain colour. As well as the perceptual metrics, MI indicates that images A and B are more similar than A and C.

5.1.5.4 Noise introduction



(a) Source image (b) Noise image

Figure 5.6: Inputs for our noise test

In this final test we conducted we took an image, Figure 5.6(a), and some random (additive) noise, Figure 5.6(b). We scaled the noise between 0 and 1, where 0 would have no effect, and 1 would potentially dominate every pixel. Having done this we compared each of the noisy images to the original image. We further normalised the output of the metrics. Clearly we expect that with no noise the metrics would return 0 as the distance, since we are after all comparing an image to itself.

Again, in Figure 5.7 we can see a similar pattern to what we have seen in

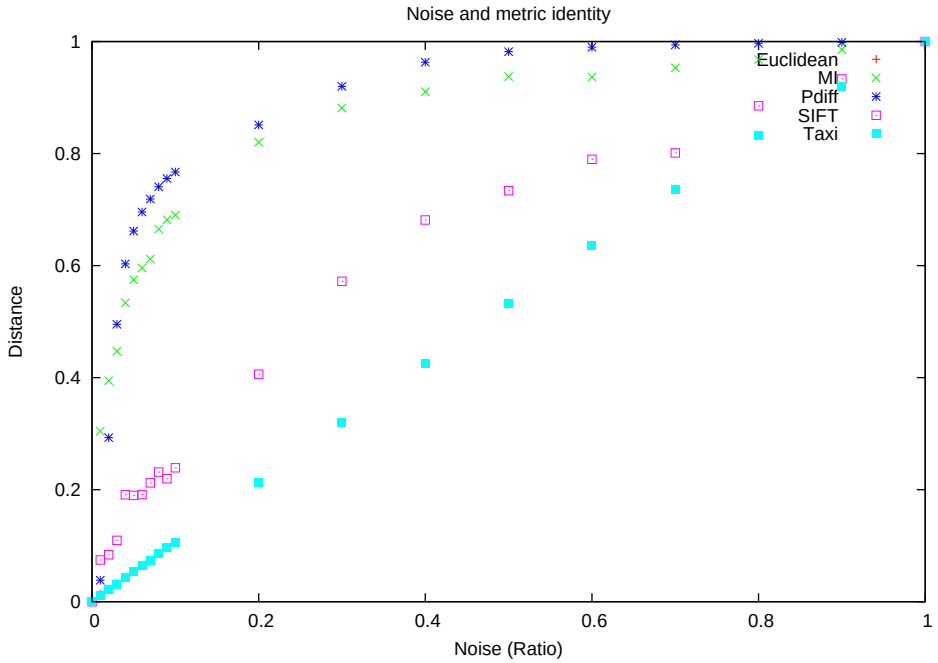
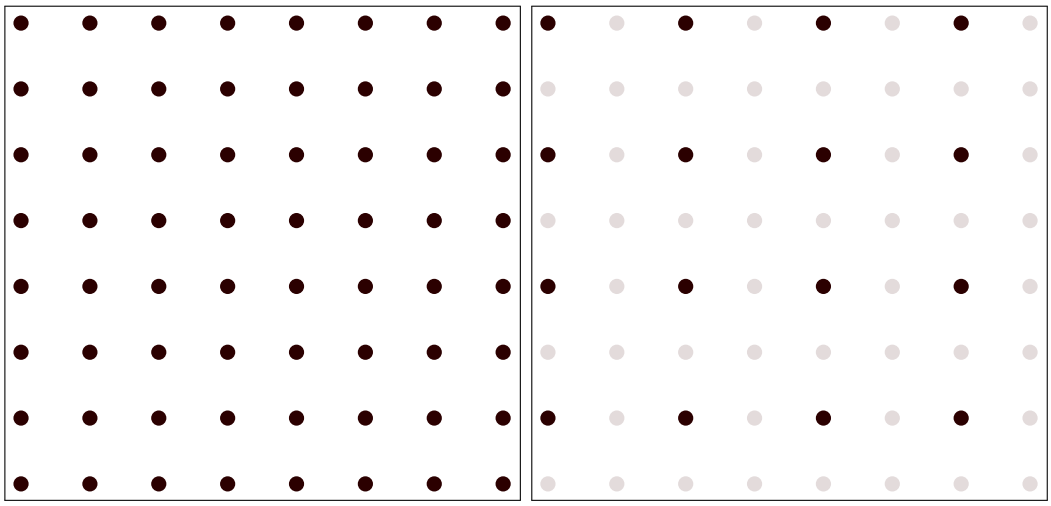


Figure 5.7: The effect of adding noise to an image and comparing with the original image.

previous tests of the metrics. Firstly the (normalised) Euclidean and taxi metrics almost exactly follow each other and are much less sensitive to noise than the other metrics. Furthermore we can observe that MI and pdiff have followed a similar (although not quite identical) pattern — pdiff peaks earlier than MI, but this could be accounted for by the fixed parameters such as viewing distance in pdiff. As with previous results SIFT shows small local fluctuations, but follows the general trend again.

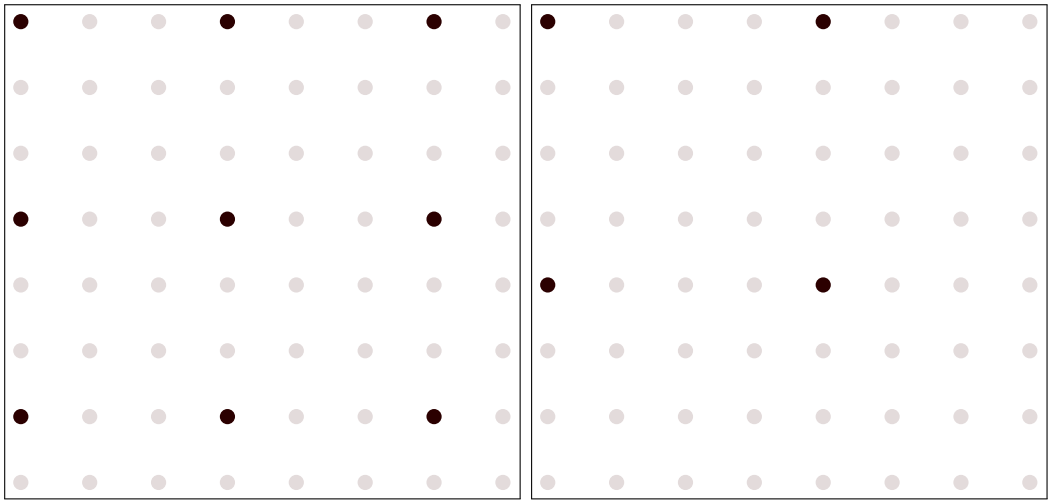
5.2 Experimental setup

In addressing our aims, one of the objectives was "Investigate possible methods of representing manifolds embedded within image space". We propose to use only a reduced set of samples in the construction of our models. Topologically at least, all of our datasets are sampled as an n-D grid in the parameter space, even if in practice knowing a position on this grid does not correspond to knowing an



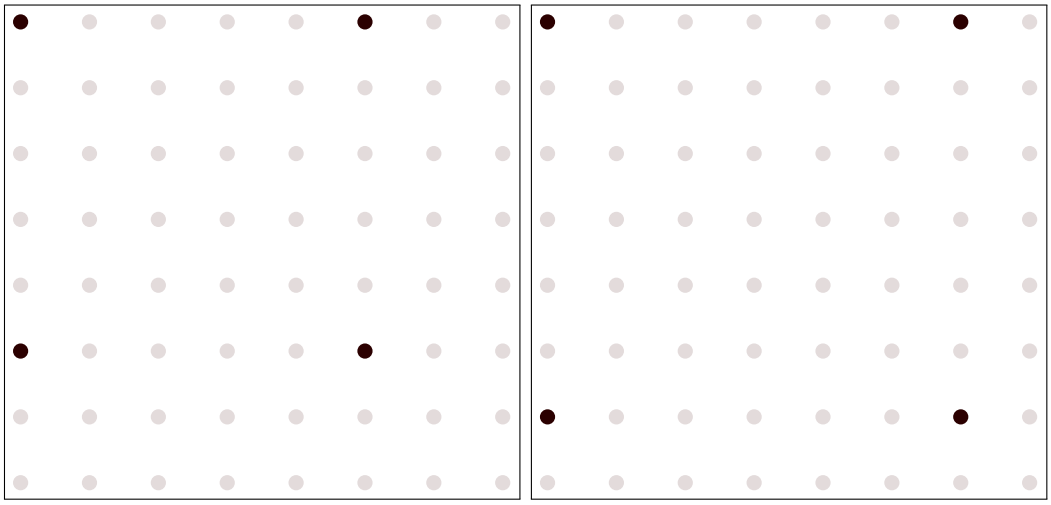
(a) The original grid

(b) The grid down-sampled by 1



(c) The grid down-sampled by 2

(d) The grid down-sampled by 3



(e) The grid down-sampled by 4

(f) The grid down-sampled by 5

Figure 5.8: Downsampling a manifold sampled on a grid

actual position in parameter space. An illustration of our down-sampling, with the terminology we use throughout, is shown in Figure 5.8. In this example we have a 2-D grid, however our down-sampling is not dependent on the dimensionality of the grid itself. Similarly, generalised down-sampling will be used for any grid. In total we will be discarding $(g+2)^d - 2^d$ samples, were g is the down-sampling number, and d is the dimensionality of the grid we are performing it on.

We have deliberately avoided using any up-sampling methods here in order to ensure that the output we observe from our models cannot have been influenced by any interpolation performed during up-sampling. This means that in many experiments we will be unable to use a number of the samples for the manifold purely because they lie outside of the down-sampled grid.

In practice we will not consider every single downs-sampling factor between 1 and the width of the grid, simply because the results from 21 and 22 will inevitably be similar, and the additional computation that would be required would be prohibitively expensive. Similarly we will not consider working in samples which get dropped because they are outside of the down-sampled grid by shifting the starting point either. We will instead consider (where there are sufficient initial samples) the following down-samplings: 2, 5, 10, 20, 25, 50, 100.

It is hoped that such down-sampling will be sufficiently representative of the general trends which would be observed were we to conduct the experiments with all possible down-samplings.

Many of the manifolds we have and plan to study are quite high resolution. Given that, as with any image based rendering technique, runtimes are proportional to the size of the input/output images in question, we will run our experiments initially at a low resolution. This low resolution output will be 50×50 pixel images, sufficient that they can be viewed and are still discernible as images, but small enough as to significantly reduce run times. This choice of low resolution experiments need not be limiting, we propose to use the low resolution results to

guide selection of a smaller set of high resolution experiments to run. Furthermore, (Lu, 1998) in his thesis argues that "the sampling process is a geometric invariant, specifically, two manifolds sampled at different resolutions are isometric (i.e., have the same size and shape)."

5.3 Conclusions

It is widely accepted that no single metric is ideal even for a single class of problems, let alone superior to the rest for all classes of problems. In the preceding sections we have attempted to address our own choice of metric for this work. Here we do not attempt to argue that anyone metric will be superior, even for evaluating our own work. Instead we have presented a number of metrics from the literature, as well as proposing one of our own, which we believe can each contribute to the evaluation of our results in subsequent chapters. The properties we have observed and discussed here will become more relevant and important further on, when we start using some of the metrics presented here to evaluate the output of the models we construct.

For now however we must accept that we are caught in something of a 'chicken and egg' situation — we do not currently know of an intrinsic metric for any of our manifolds and, given that we only have a very limited number of samples of all of our manifolds, developing such a metric is not really feasible. Furthermore we do not have a metric which can identify how far off the manifold an image is.

Given the scale of images our proposed experiments will generate it is clear that automated evaluation of some form is vital, and furthermore that visualisation of the results is likely to be an important issue. We will therefore be devoting more time in subsequent chapters to addressing visualisation of image manifolds.

5.4 Summary

In this chapter we have seen:

- an overview of several image metrics. We will use this knowledge to facilitate our evaluations in future endeavours;
- we propose that mappings of SIFT keypoints represents an approximation of a perceptual metric;
- an experimental methodology that allows for principled, automated experiments with large volumes of images.

In the next chapter we will explore Mid-points and metrics further to study the local shape of our image manifolds.

Part II

Looking at image manifolds

Chapter 6

Measuring image manifolds

This chapter introduces two methods for measuring properties of image manifolds in general. The goal of this chapter is to demonstrate that it is possible to measure, or at least approximate, useful and interesting properties of image manifolds from the discrete samplings we have available. At this stage these measurements are introduced as an interesting point of discussion, and results are analysed and discussed in this chapter. Later in this thesis (Chapter 12) we refer back to one of these measures and show as a proof of concept how we can use it to improve the construction of our models.

Both of the measurement methods we introduce here compute properties of the manifold that are local to a given point on the manifold. Similarly both of the techniques are designed, implemented and discussed from the perspective of a manifold with intrinsic dimensionality of 1, embedded in full image space. We deliberately chose to do this, and a simple, natural, generalisation of this to the n-D case can be produced by considering the value at a given point to be some combination of the values in each of the n dimensions. This could be as simple as a sum, however there are a number of potential problems (e.g. high curvature in a direction which is not aligned with an axis) with a simplistic approach like this and more complex definitions of curvature from differential geometry (Spivak, 1979) could be considered.

6.1 Midpoint distance

Our image manifolds are sparsely sampled, and the behaviour of the manifold between samples is uncertain. It is possible however to use a variety of techniques to at least inspect the structure of an image manifold and learn more about image manifolds in general. Discussed here is the first of two procedures we used to inspect the non-straightness of a few specific image manifolds, which we term *midpoint distance* and refer to as MD.

Key to our measurements is the concept of a midpoint (Berard, 1971/72). We previously introduced the concept of a metric in Chapter 5. Given a metric space (X, d), a point $m \in X$ is said to be a midpoint of $x, y \in X$ if $d(x, m) = d(y, m) = \frac{1}{2}d(x, y)$. It becomes apparent that some metrics will measure distance between points in such a way as to produce midpoints which do not themselves lie on the manifold. This gives rise however, to a class of distance metric, known as intrinsic metrics, which do measure distances on a manifold. Finding an intrinsic metric for all but the most trivial of image manifolds is far from straightforward and we have seen in Chapter 5 that none of the proposed metrics are intrinsic.

What we do know though is that if our manifold is parameterised by, say, a rotation of an object, then when we later construct models what we want (Chapter 4) is a model such that the image that represents a 5° rotation is halfway, on the manifold, between 0° and 10°, regardless of the actual distance in image space. Further, we can say that if the manifold was a straight line, traversed at constant speed, then the midpoint would indeed be the Euclidean midpoint. Since by calculation we know the Euclidean midpoint, and by selecting neighbouring samples on the manifold we also know the real midpoint on the manifold, it is possible to measure the Euclidean distance between these two points (known and calculated midpoints) in image space. The Euclidean midpoint defines the midpoint of a uniformly traversed straight line in image space, it follows therefore that the distance between this midpoint and the known one can be seen as a measure of

deviation from straightness present in the image manifold. The definition of 'non-straightness' we introduce here deliberately explicitly considers both the position of the samples and the speed of traversal of the manifold itself, because the goal is to accurately model both of these when we later come to construct models of these image manifolds.

Formally, given a sequence of uniformly sampled, in *parameter space*, images $\mathbf{I}_n$ which form a 1-D manifold, we define the local deviation from straightness, S_n, at a given point, n, as

$$S_n = D(\mathbf{M}_n, \mathbf{I}_n), \tag{6.1}$$

where $\mathbf{M}_n$ is the midpoint of the two neighbouring images, $\mathbf{M}_n = \frac{\mathbf{I}_{n-1}+\mathbf{I}_{n+1}}{2}$ and $D(a, b)$ is the (Euclidean) distance between two images. We can further generalise this by considering not just the immediate neighbours, but the neighbours some distance (in terms of the *number of* sampled images) g on either side of the point in question. We therefore modify the original definition to:

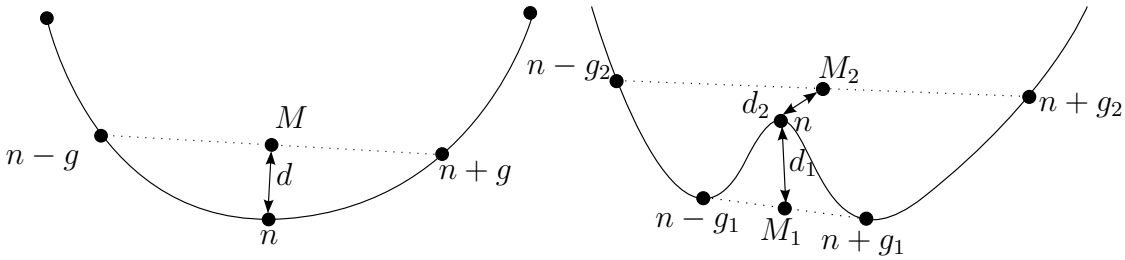
$$\mathbf{M}_{n,g} = \frac{\mathbf{I}_{n-g} + \mathbf{I}_{n+g}}{2}, \tag{6.2}$$

$$S_{n,g} = D(\mathbf{M}_{n,g}, \mathbf{I}_n). \tag{6.3}$$

In this way we can consider the non-straightness of our manifolds at coarser scales than they were originally sampled at, which potentially may reveal more interesting information than just considering very small distances.

This measure is illustrated in Figure 6.1(a), which shows an hypothetical image manifold. This example demonstrates how we measure the distance between the real and generated midpoints.

This method considers all possible values of n, the image index along the manifold, and every value of a gap g where both $n + g$ and $n - g$ are valid image indices within the bounds of the manifold we have sampled. In the instances



(a) Measuring the distance, D between real and generated midpoints

(b) An example where curvature appears to decrease when increasing g

Figure 6.1: Measuring deviations from straightness

where the manifold is periodic the indices are wrapped around. As a result of considering every possible midpoint and every possible gap value we are able to see many details in our results about the image manifolds at various different scales of observation.

Measurements taken like this rely upon several assumptions. Firstly, we must assume that the change between samples is small, in fact we must assume we have sufficient samples to accurately capture at least the main features of the manifold to begin with, as illustrated in Figure 6.1(b). Secondly, this method will at best ignore extremely small features such as small corrugated parts of a surface, and in the worst case confuse results. Thirdly, because the measure used relies upon the notion of distance, the metric chosen has considerable impact upon the non-straightness measured. In Chapter 5 we introduced a number of different metrics. At this point however we have only discussed non-straightness in terms of the Euclidean metric. This is intentional, and in this chapter we will only be using the Euclidean metric, because computing midpoints for other metrics is a significantly more costly process (i.e. would have to be formulated as a minimisation problem) and potentially there may well be more than one candidate midpoint. This is however an avenue of research we hope to be able to follow up in the future.

6.2 Curvature

The midpoint distance introduced previously is entirely geometric and not founded on any formal definition of curvature. This is not a problem and our results in Section 6.3 show there is indeed a strong relationship between features in the images and the reported values. Nonetheless we introduce here a second, more formal measurement on the image manifolds.

This second, potentially interesting property of an image manifold is that of its curvature. The curvature of a manifold has implications for any attempts to sample or model it, including the accuracy of applications explicitly or implicitly making assumptions about image manifolds. In this section we discuss a method of determining the curvature of a simple image manifold.

There are a number of theoretical measures of curvature, for instance (Xu and Bajaj, 2003; Zhang and Xu, 2007). These however are not practical to compute for the real-world manifolds in which we are primarily interested. Instead, the process we use here is to formulate a discrete approximation, which is the same as used in (Lu, 1998, Section 4.2.1).

The curvature k along a path $\mathcal{P}$ is defined as the derivative of the tangent t with respect to arc length s:

$$k = \left|\left|\frac{\partial t}{\partial s}\right|\right| = \left|\left|\frac{\partial^2 \mathcal{P}}{\partial s^2}\right|\right|. \tag{6.4}$$

Following Lu (1998), from the definition in Equation 6.4, a numerical approximation can be defined as follows:

$$k \approx \left|\left|\frac{\Delta t}{\Delta s}\right|\right|, \tag{6.5}$$

with t as the unit tangent vector. Given the path parameterised by $x = x(v)$, x_j defined as $x_j = x(j + g)$ and a span u_j given by $u_j = x_j - x_{j-1}$ then the unit

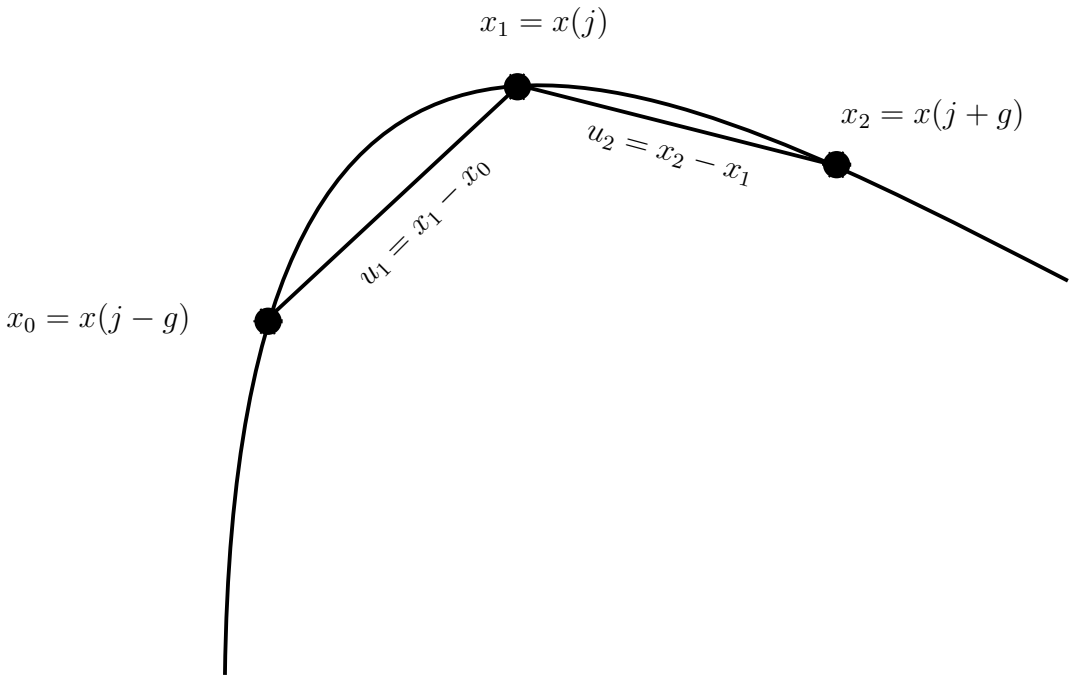


Figure 6.2: The approximation of path curvature of Lu (1998)

tangent at j can be approximated by $t_j \approx \frac{u_j}{||u_j||}$. If the change in arc length over a span g is $\Delta s \approx ||u_1|| \approx ||u_2||$ then it is possible to write Equation 6.4 as the limit

$$k = \lim_{g \to 0} \left| \left| \frac{\frac{u_2}{||u_2||} - \frac{u_1}{||u_1||}}{||u_1||} \right| \right|. \tag{6.6}$$

This is illustrated in Figure 6.2. Later we discard the limit in Eqn. 6.6 and consider various value of g alongside the midpoint distance.

Lu (1998) points out that there are a number of potential problems with this. Firstly, as with the midpoint distance measure discussed previously, the choice of g heavily influences the results of the approximation. Small values of g will suffer from resolution problems and quantisation problems. Large values of g will introduce aliasing problems. Secondly, this approximation is also badly affected by noisy data. To work around these problems Lu (1998) introduces a further method for approximating curvature of a 1-D image manifold, by fitting a quadratic, presented as a minimisation problem. We deliberately choose to avoid this method here, because our eventual goal is the fitting of curves and surfaces to the data

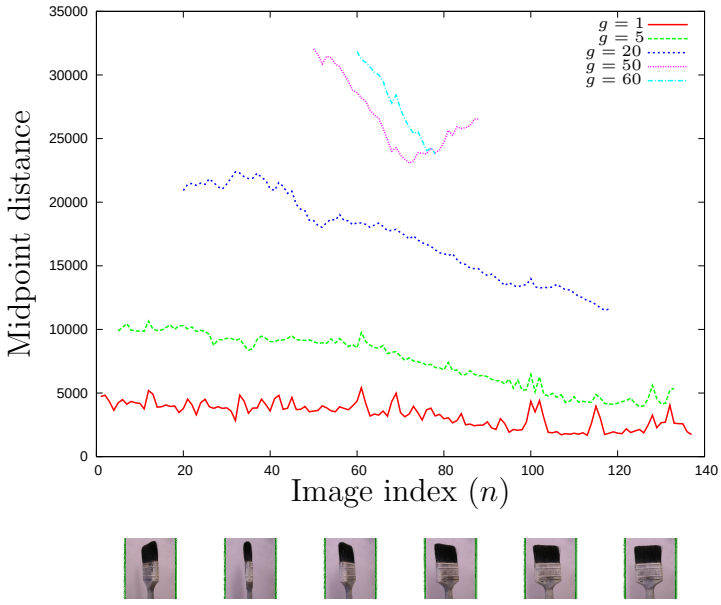


Figure 6.3: Midpoint distance measure of the BRUSH image manifold

itself. By fitting any model to the data at this stage to estimate curvature we risk unduly influencing our results and creating a *circulus in probando*.

6.3 Results

In this instance we look at the two proposed measures using four selected image manifolds. Of these, only one (TEAPOT) is periodic, and thus for larger values of g results are only available towards the centre of the manifold for the others. The results for the midpoint distance from the sequences considered are presented in Figures 6.3, 6.4, 6.5 and 6.6. In these figures we have elected to show only a limited number of g values as showing every possible g value on a single graph makes it difficult to interpret the results. The values of g are selected to represent the more noteworthy elements of the results.

The midpoint difference measured in the image manifolds has a clear relationship with the geometry of the objects being studied. This is seen most clearly with the STRAIGHT_8 sequence, Figure 6.4, where the main peak seen at around $n = 127$ corresponds exactly with the robot passing a red box on its route, when

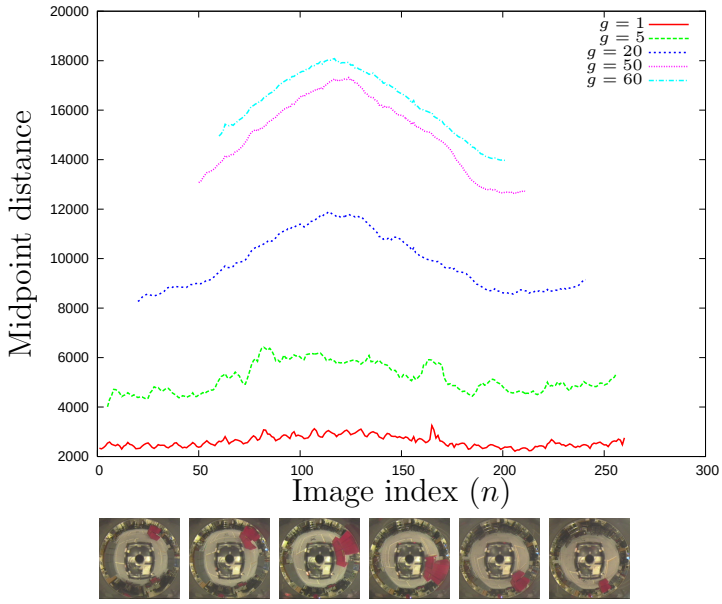


Figure 6.4: Midpoint distance measure of the STRAIGHT_8 image manifold

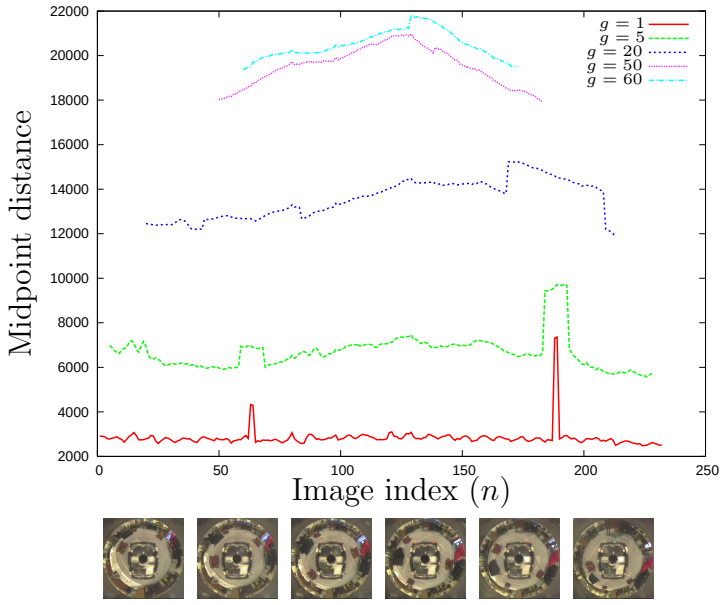


Figure 6.5: Midpoint distance measure of the STRAIGHT_1 image manifold

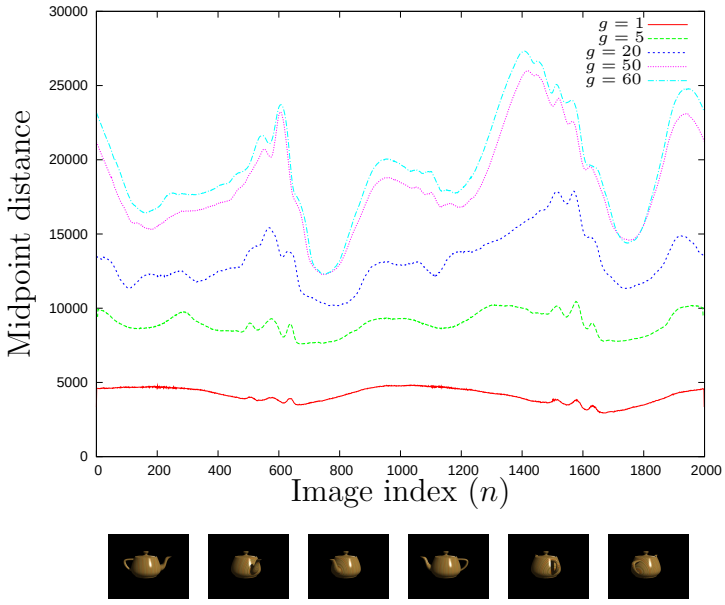


Figure 6.6: Midpoint distance measure of the TEAPOT image manifold

the changes between images are most dramatic. It is also noticeable in the paint-brush manifold, Figure 6.3, where the manifold is considerably less straight at the point $n = 38$ where the large face of the brush passes back into sight of the camera.

It is interesting to see how in Figure 6.3 the $g = 60$ line actually intersects the $g = 50$ line at around the $n = 80$ point, implying that at this length we have hit the situation illustrated in Figure 6.1(b). A similar behaviour is observed in Figure 6.6 at points $n = 750$, and $n = 1700$. It is likely that this behaviour in the TEAPOT image manifold is caused by the regular texture applied to the teapot. This may cause it to display the "corrugated" manifold surface effect illustrated in Figure 6.1(b) as soon as g is larger to span more images than make up one repetition of the elements in the texture.

The presence of what appears to be a regular pattern of almost constant frequency when $g = 1$ in Figure 6.4 and Figure 6.5 was slightly unexpected. After further investigations using position data captured while the robot was moving during the capture of the images, it would appear that this behaviour is caused by the controller in the robot. The cause of the anomalous points at $n = 166$ in

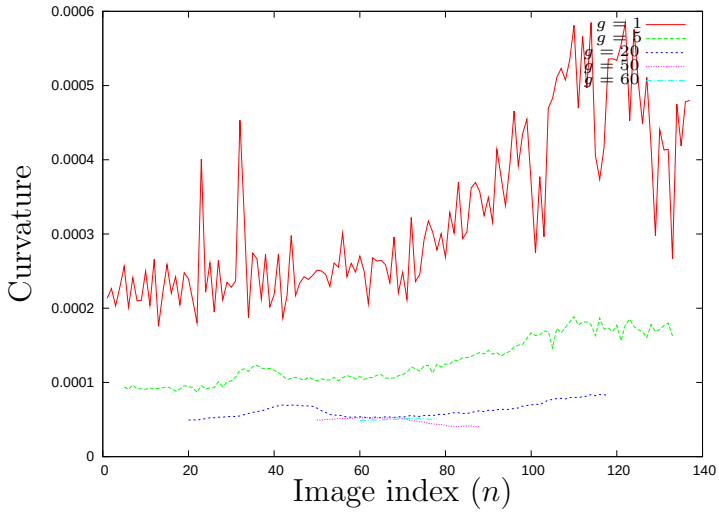


Figure 6.7: Curvature measure of the BRUSH image manifold

STRAIGHT_8 and $n = 64$, $n = 189$ in STRAIGHT_1 are known to be caused by missing frames in the video sequence, caused by the capture system.

It also would appear from all the results shown here that the rate at which the measured midpoint distance increases as the gap considered increases is converging to some finite limit or maximum distance for each image manifold.

Results from the curvature measure are presented for the same four datasets in Figures 6.7, 6.8, 6.9 and 6.10. The most obvious observation from these is the more notable appearance of high frequency variations in the curvature reported, in all the examples except TEAPOT (which does seem to show similar features at a much smaller amplitude). This would appear to indicate the presence of noise in the images used. In general we can see the same trends present in both midpoint distance and curvature, however they appear to be inverted. This would seem to confirm again that what we are observing is indeed a result of the images we are considering and not simply an artifact of the measurement used.

6.4 Discussion

What does this show us about the image manifolds we have studied? Firstly we can see, as one would expect, that both the midpoint distance and curvature of

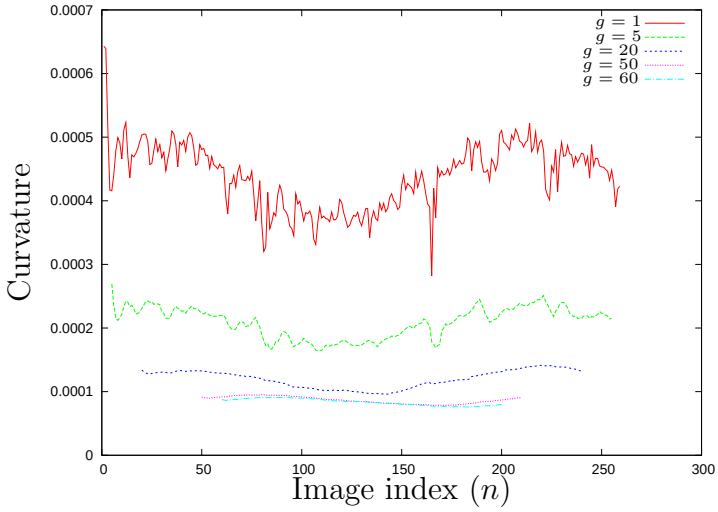


Figure 6.8: Curvature measure of the STRAIGHT_8 image manifold

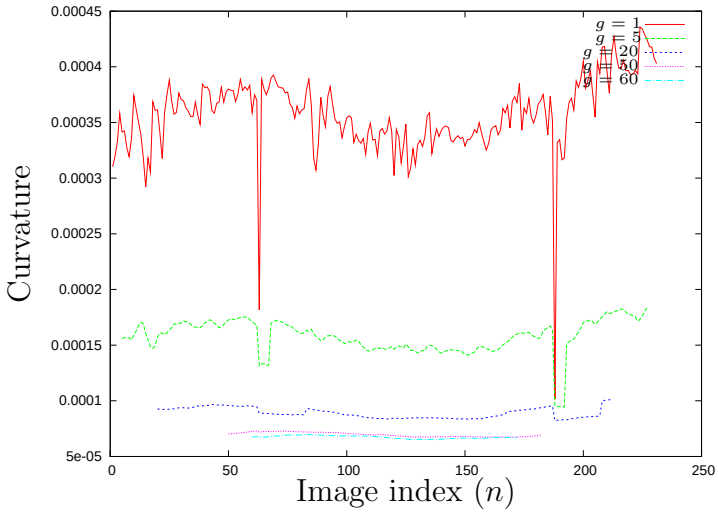


Figure 6.9: Curvature measure of the STRAIGHT_1 image manifold

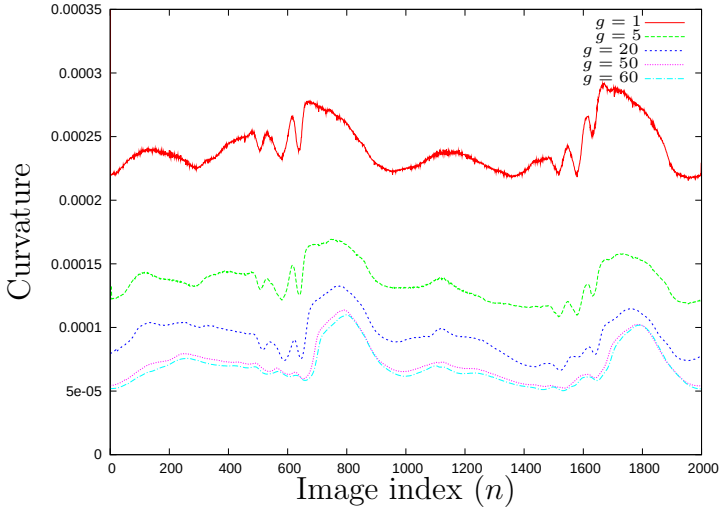
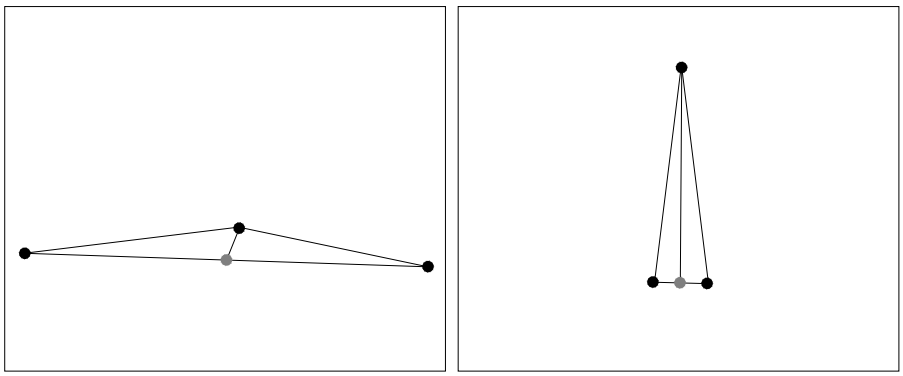
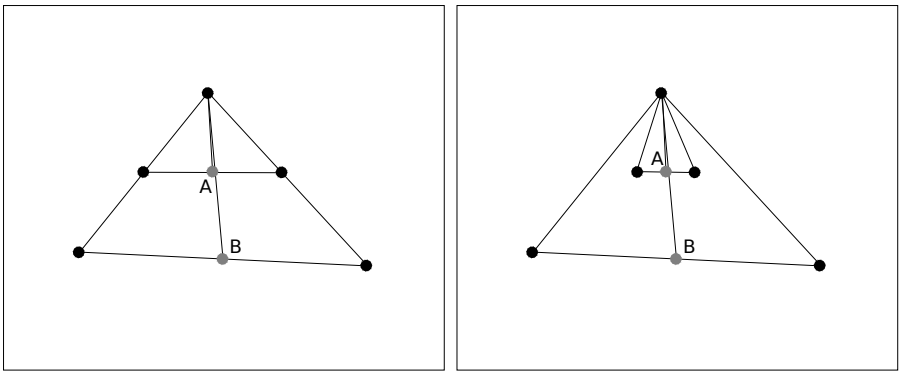


Figure 6.10: Curvature measure of the TEAPOT image manifold



(a) Both midpoint distance and curvature report low values

(b) Both midpoint distance and curvature report high values



(c) B compared to A, midpoint distance reports a higher value, curvature by comparison is lower

(d) B compared to A, midpoint distance reports a higher value, curvature reports a lower value

Figure 6.11: Comparing midpoint distance and curvature. The black dots represent sampled images, the light grey dots represent Euclidean midpoints.

an image manifold are not constant across its entire length. This clearly confirms the natural assumption that for sections of an image manifold where more details are changing more rapidly, more data samples will be needed to accurately sample the shape of the manifold. That is evident from considering the implications of the Nyquist-Shannon sampling theorem. The results presented thus far could be considered to be a simplification of frequency analysis of the signal that is an image manifold.

The results from the midpoint distance and path curvature method side by side show that in all cases, when g increases the midpoint distance increases whilst the curvature decreases. From the point of view of an interpolation this indicates midpoint distance is more useful than curvature.

The second fact we observe from the results is that, for a given g, when the curvature increases, the MD decreases. To explain this we need to consider a number of different cases of shape and size for the triangles the midpoint distance and curvature measures relate to. Figure 6.11 illustrates a number of hypothetical scenarios, which we will consider for both midpoint distance and curvature measures. Note that in these figures the scale is assumed to be the same between all triangle, and for clarity where more than one triangle is shown they are superimposed. In reality, for a fixed g value, the triangles would typically either share edges, or even be completely separate in image space. Figures 6.11(a) and 6.11(b), where the length of the two edges of the triangles are the same, show examples of scenarios where, midpoint distance and curvature will change in the same direction. On the other hand what is illustrated in Figures 6.11(c) and 6.11(d) are two scenarios whereby as the midpoint distance increases the reported curvature will decrease. This occurs because when we apply Equation 6.6 the only change between triangles A and B in Figure 6.11(c) is in the denominator. This tendency of the curvature measure to understate the effects of large changes in image space because of the normalisation term makes it less suitable for controlling the sampling of image manifolds in later work. From Equation 6.6 there are two possible ways to make the curvature large, either by causing the numerator itself to be large (i.e. large change in tangent orientation), or make the denominator small (i.e. close points in image space). The scenario in Figure 6.11(d) shows both the numerator and the denominator changing to decrease the reported curvature at a point when the midpoint distance is increasing.

We do not encounter scenarios similar to Figure 6.11(a) and 6.11(b) and in practice the distances between midpoints and known images varies similarly to the distance between the images themselves. Overall the image manifolds are not very straight (we see almost no evidence of Figure 6.11(a) occurring) and similarly not excessively curved either (we see little evidence of Figure 6.11(b) happening

either).

This observation seems to fit when considering the ratio of 'general background' to foreground in the images and the corresponding effect this has on the percentage of pixels which change between successive images compared to how much they change. In the case of the BRUSH when $n \approx 110$ the brush is almost directly facing the camera. At this point a small change in the rotation of the brush will cause a large percentage of the pixels in the image to change. Of the pixels that do change however the changes will, generally, be comparatively small. This seems to imply that we are probably seeing a change similar to moving from triangle B to triangle A in Figure 6.11(d). Pixels that were part of the bristles are largely black and will continue to be so in neighbouring images. Likewise for the handle of the brush. The results reported here indicate that whilst the changes themselves are comparatively small (as reported by MD) the changes in tangent orientation are comparatively large, implying that locally there is a highly curved, complex structure in image space[1]. The same argument applies directly to the TEAPOT dataset. Both of these datasets have comparatively static backgrounds although in the case of the BRUSH there are some fairly significant changes caused by illumination changes from the movement of the sun and the camera itself during acquisition.

In the case of the STRAIGHT_8 and STRAIGHT_1 datasets, in the majority of frames the background itself is the dominant feature (i.e. occupies the majority of the pixels). When not near the boxes these change from frame to frame, typically not by large values, but enough to cause the change in normalised tangent orientation to vary considerably between consecutive images. When passing by the boxes placed in the lab however this changes: many more pixels are red and consistently red between frames. It appears then that this causes the change in normalised tangent orientation measured to reduce.

[1]The implication is that this can be better (in the Euclidean sense) approximated by a straight line than the areas where the tangent varies less.

It is also interesting to consider the implications of the human visual system (HVS) in all of this — after all any attempt to use an image manifold for view synthesis will potentially be viewed and judged by humans. The HVS has many interesting characteristics which affect perception of images, which are covered in quite some depth in the literature (Daly, 1993; Neumann et al., 1997; Pappas and Safranek, 2000; Ramasubramanian et al., 1999). What is common amongst all models of the HVS is that there will be some differences which are imperceptible to humans viewing them. This means that we can significantly reduce the sampling of our image manifold if our aim is weakened to less than 100% accurate image reconstruction, but still keep inaccuracies imperceivably small. In fact this knowledge of the HVS has already been exploited successfully in many lossy image compression schemes.

The non-constant curvature and midpoint distances across the sequences means that for certain segments of the manifold we may be able to discard significantly more sample points and produce visually accurate replacements using (image space) interpolation with a given metric space, from the discarded points' neighbours. Our motivation for this is to enable simplifications in terms of storage requirements for image manifolds. Keeping the number of sample points low without noticeably sacrificing image quality helps to make other applications of image manifolds more feasible.

Intuitively the notion of midpoint distance used is more closely connected with the notion of the human visual system. Humans are adept at discerning changes in images based upon the magnitude of the change itself and not the rate of change of the tangent in image space. We therefore propose that when we come to use measures of the manifold later on that the midpoint distance measure is a more suitable choice for use in such applications.

We may be able to suggest from this notion of non-straightness a reasonable limit for the maximum midpoint distance of a manifold segment before represent-

(a) Midpoint image $g = 5$, $n = 5$.

(b) Midpoint image $g = 1$, $n = 5$

(c) Midpoint image $g = 5$, $n = 105$.

Figure 6.12: Examples of change in midpoint distance across the paintbrush image set. Clear visual evidence that some parts of the sequence, can be better approximated to a straight line than other parts

ing it as just two images looks noticeably inaccurate. Figure 6.12 shows some anecdotal evidence for a reasonable limit to guide this sampling. Notice in particular the difference in quality between the two images Figure 6.12(a) and Figure 6.12(c) both of which represent interpolation between two images the same distance apart in terms of number of samples. Also note that Figure 6.12(a) and Figure 6.12(b) are both the same image index in our sequence.

The limit at which the visual quality reduction becomes noticeable is a very

subjective notion, which is further affected by many factors including the viewing distance, the age of the viewer and the resolution, with resolution being the only one of these we are able to control completely.

6.5 Conclusions

The work presented here, in particular the suggestion that there is a distance at which images become noticeably degraded is of particular interest to this thesis. Additionally it would be interesting to apply the limits suggested to algorithms that currently use arbitrarily selected cut-off points for maximum permissible errors. The results presented here show that for the example manifolds we have considered both the midpoint distance and the curvature is non-uniform, and dictated by the content of the images themselves. In Chapter 12 we will return to the midpoint distance and demonstrate how it may be applied to improve the construction of models of image manifolds.

Whilst it is possible the changes we have observed in a results have come about simply because of noise this seems to be unlikely. Analysis of the shapes of the triangles (Figure 6.11) seems to imply that what we are observing is far more regular and structured than what would be produced by noise alone. Investigating these structures further is an interesting area for future work outside of the scope of this thesis, but discussed briefly in Section 13.3.

As stated previously we have only shown results from the 1-D case, by way of example, and suggested a possible simple extension to higher dimension surfaces. It is interesting to note that Lu (1998) also dismisses notions of curvature from differential geometry in favour of numeric approximations. Lu (1998) avoids problems with high curvature not aligned to a particular axis by studying the curvature of particular paths on his manifolds, as well as along the axes. Given that for our purposes the curvature measurement is not useful, combined with the concern about the influence of noise on second order derivatives, we will not be

considering it further (Chapter 12). Therefore the lack of a formal definition for higher dimensions or study in higher dimension is not unduly limiting.

In this instance we have only considered the Euclidean metric for midpoint distance but other metrics and pseudo-metrics do exist and it would be interesting to extend this study to cover these. Unfortunately for a number of the metrics we introduced in Chapter 5 calculating midpoints remains at best an expensive directed search in image space, with no guarantee of a unique midpoint.

6.6 Summary

In this chapter we have seen:

- a geometric method of estimating the non-straightness of image manifolds;
- a discrete approximation of the curvature of the image manifolds;
- some results from the application of these to a number of image manifolds.

In the next chapter we will:

- look at dimensionality reduction techniques for our image manifolds;
- use this to devise a visualisation strategy for image manifolds and later results.

Chapter 7

Dimensionality and Visualisation

7.1 Introduction

We have already seen Principal Component Analysis (PCA), which can be thought of as finding vectors which explain the variance of a dataset, put to use in the context of image manifolds in Chapter 3. This was in the form of the 'Eigenfaces' of (Bichsel and Pentland, 1994; Turk and Pentland, 1991*a*,*b*), whereby PCA was applied to the high dimensional space of a set of face images in order to select a lower dimensional subspace that recognition and matching may be performed in. The results of this were good, both numerically and visually, as a result of which PCA has become an established technique for computer vision problems, being applied to many more recognition problems than just face recognition. Almost any recognition or machine learning problem which can be formulated as a high dimensional feature vector, of which images are just one example, can be approached in this way.

Earlier, in Chapter 6, we also discussed the curvature of image manifolds, and noted that in all but the most trivial of cases image manifolds tend to have very high curvature and be non-linear, which makes techniques like PCA unsuitable for modelling them. In the case where the intended application is that of recognition the loss of detail this would entail is more than acceptable — a general loss of

detail is often a good thing in the case of visual recognition, given that a scene or object is unlikely to be identical between any two images of the same object.

This is not, however the end of the road for PCA with image manifolds in this work. A major problem when working with image manifolds is the question of how to visualise such a structure, given the high dimensionality of the space within which it is embedded. Clearly for all but the most simple (1×3 pixel grayscale) images direct visualisation of the image space is non-trivial. Visualisation is however, a potentially very useful tool in the study of image manifolds, and the enormous loss of information inherent to any slicing (or similar) performed directly on the image manifolds is deeply troublesome. We have no way of knowing how or where to take a slice, nor do we have any way of knowing how representative of the image manifold as a whole any slice is. Furthermore traditional visualisation techniques which would be used in a scenario like this, such as encoding using glyphs, colour, multiple views or interactivity, are barely applicable given the sheer size of the image space in which the manifold is embedded.

Given these problems we therefore seek a principled approach in the selection of a suitable subspace in which to view our image manifolds. Having seen how PCA has been successfully applied in other vision problems, as well as its general applicability for clustering and dimensionality reduction, we therefore elect to study it further, using it as the basis for a visualisation technique for this work. In this vein, throughout the remainder of this thesis, we apply PCA, as was also done in (Nayar et al., 1996), to select a new coordinate system into which we project individual points or structures from the high dimensional image space prior to visualisation.

Whilst this chapter does not introduce anything novel it is important that PCA as a visualisation technique is discussed and introduced prior to our adoption of it in the remaining chapters of this thesis. Furthermore this chapter addresses some of the important details of this technique, as well as presenting an overview of

related previous work. We conclude this chapter with a presentation and discussion of some of the results obtained. Critically these results lend further credence to the concept of an image manifold. Full details of our results for all the datasets are presented in Appendix D.

7.2 Implementation

PCA is a topic well covered by most statistics textbooks, e.g. (Kendall and Stuard, 1968), usually located in a chapter on multivariate analysis. Informally PCA can be thought of as identifying vectors which best explain the variance within a given dataset and is a helpful high level perspective from which to view things.

In practice there are a number of possible ways of performing PCA. Each method is likely to perform differently, both in terms of time complexity and error. From the context of this thesis and in the interest of brevity PCA is treated as a tool, and we consider two existing approaches to performing PCA.

Since performing PCA is the same as solving the eigenvalue problem with the covariance matrix of the data we can very straightforwardly use a technique like SVD, which solves the eigenvalue problem. This method is referred to as 'impca' henceforth. There are a large number of singular value decomposition algorithms available. In this instance we elected to use a form of the singular value decomposition algorithm presented in (Press et al., 2002). For more discussion on the algorithms involved the reader is referred to (Golub and Van Loan, 1989). The SVD algorithm of Press et al. (2002) essentially performs an Householder reduction and a QR decomposition.

We also consider an alternative, since it transpires that in practice computing all N^2 eigenvectors of an $N \times N$ image proves (for typical sized images) to be both slow and unnecessary. To this end we have also implemented and used the method described in (Turk and Pentland, 1991*b*) as a second approach. If we have M images in our dataset Turk and Pentland (1991*b*) propose to solve a problem

on a $M \times M$ matrix instead of an $N^2 \times N^2$ matrix[1] and from this compute only the relevant parts we are concerned with for visualisation.

We refer to this method in our results as 'mtfast' throughout, because in some circumstances it is significantly faster to compute than the full SVD, and it is much more amenable to parallelisation on multi-processor architectures.

Our datasets are all obtained off-line, consequentially methods such as (Hall et al., 1998) offer little to no advantage currently. In the future we may decide to capture and model image manifolds 'on-the-fly', in which case this may prove useful for visualisation as the model grows.

7.3 Results and Discussion

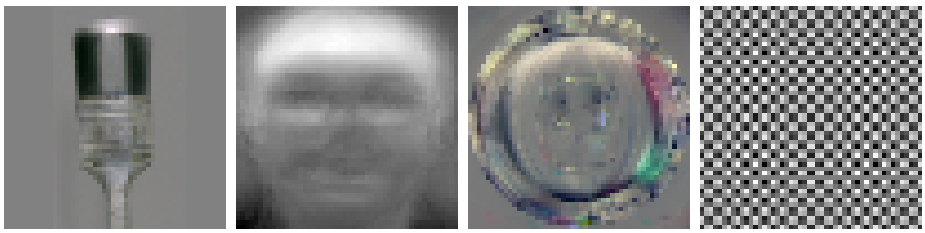
In this section we present an overview and discussion of the results of performing PCA on our datasets. In order to view the eigenvectors it is necessary to normalise them to ensure they fall within the usual $[0; 255]$ range of RGB images. In the interest of space only a subset of the results are included with the text to facilitate discussion. However full results are included in Appendix D.

Figure 7.1 shows the first principle component from four of the datasets, computed using the full singular value decomposition. As would be expected from running principle component analysis on a set of similar images the first principle component is instantly recognisable as belonging to the dataset.

From Figure 7.2, which plots the eigenvalues on a logarithmic scale, we can see how much each of the principle components identified contributes to explaining the variance within the dataset. Here it is clear that the eigenvalue of the first eigenvector is normally several orders of magnitude larger than the following eigenvalues.

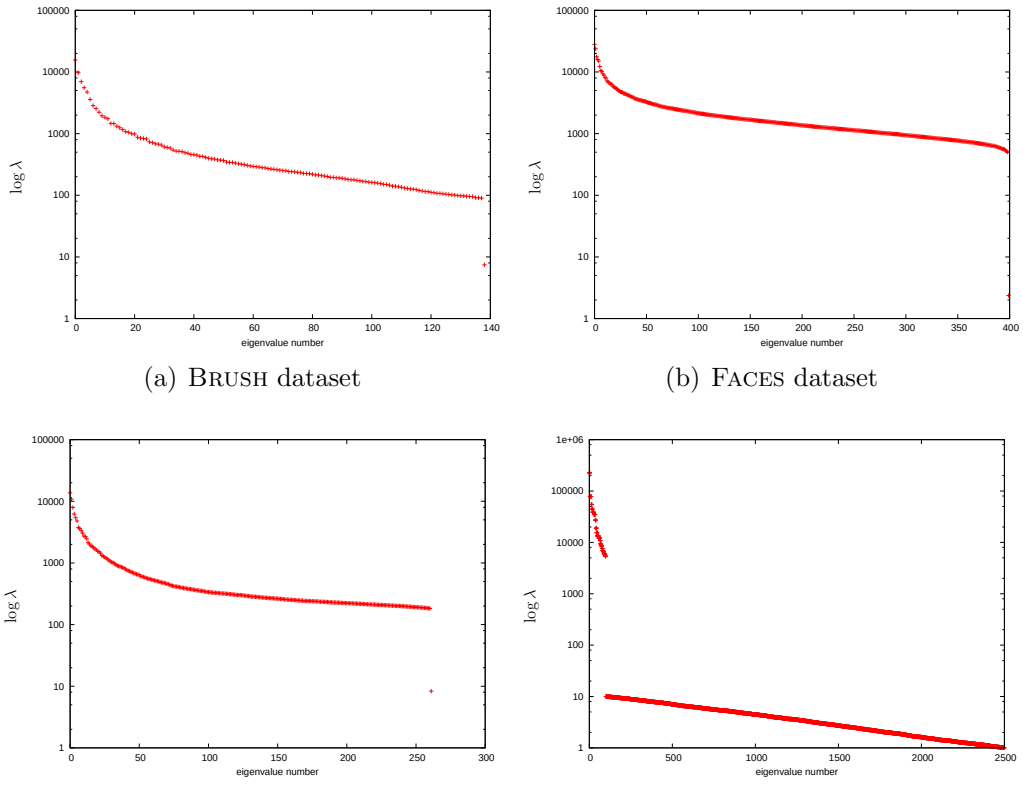
This general pattern can be observed for all our datasets. It is important

[1] For many of the lower resolution datasets, with (very) large numbers of samples this turns out to be worse!



(a) BRUSH dataset (b) FACES dataset (c) STRAIGHT_5 dataset (d) CHESSBOARD dataset

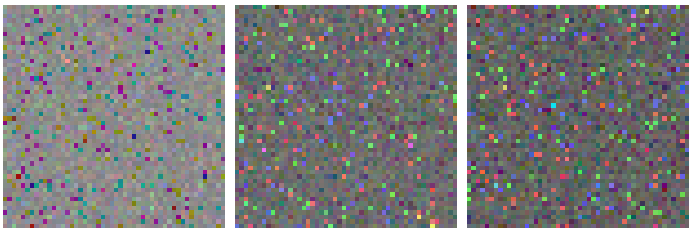
Figure 7.1: Views of the first principal component of four of our datasets, using the 'impca' method



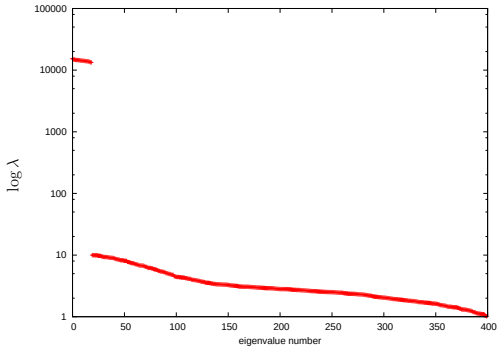
(a) BRUSH dataset (b) FACES dataset

(c) STRAIGHT_5 dataset (d) CHESSBOARD dataset

Figure 7.2: The eigenvalues from four of our datasets, using the 'impca' method



(a) First principal component (b) Second principal component (c) Third principal component



(d) Eigenvalues

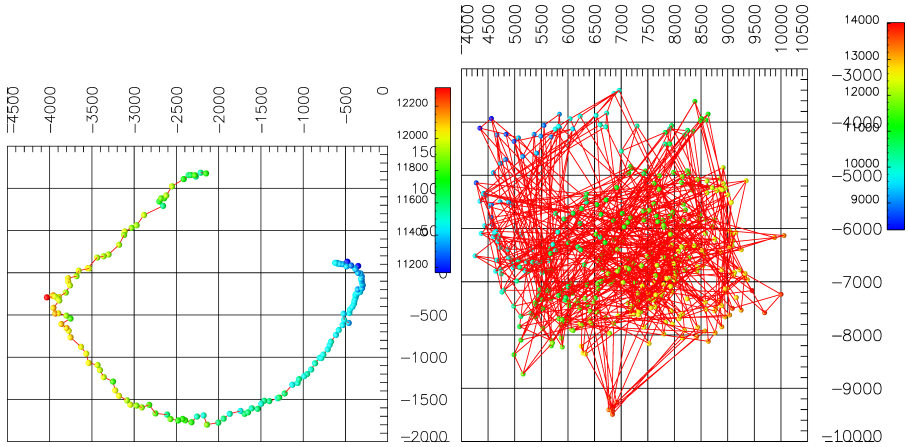
Figure 7.3: Views of the first three principal components of the BMNOISE datasets, using the 'impca' method

to note that as our eigenvalues tend towards zero the corresponding eigenvectors become increasingly influenced by the noise within the dataset.

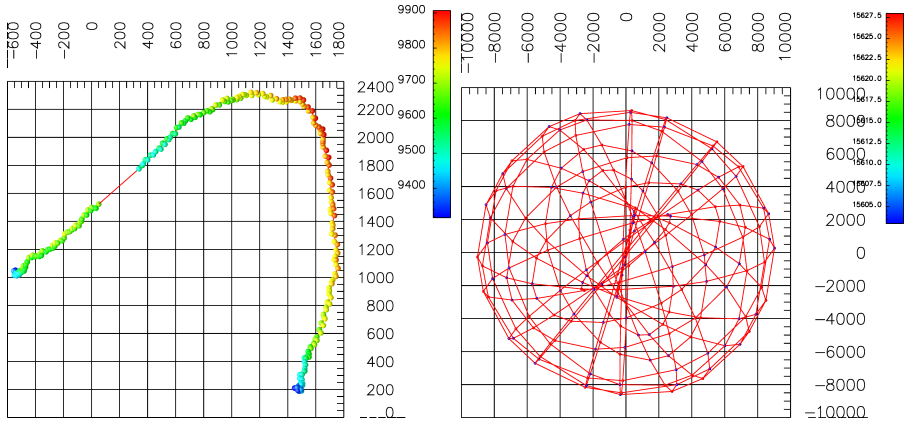
In the case of the BMNOISE dataset (illustrated in Figure 7.3) after the first eigenvalue the following 19 eigenvalues are all almost identical, indicating that they do not explain the variance any better than any other of the eigenvectors.

7.3.1 Re-projecting the manifolds

Given the eigenvectors and eigenvalues of an image manifold calculated using one of the methods outlined in Section 7.2 and having visually and numerically inspected the resulting eigenvectors, it is trivial then to re-project the image manifold onto one or more of these eigenvectors. The question remains though: which of these eigenvectors should be used? The obvious first choice would be the three eigenvectors with the highest corresponding eigenvalues since these explain between them the most possible variance that any three eigenvectors could explain.



(a) BRUSH dataset (b) FACES dataset



(c) STRAIGHT_5 dataset (d) CHESSBOARD dataset

Figure 7.4: PCA plots of some of the datasets, using the 'impca' method

A selection of these results are presented in Figure 7.4[2]. In this figure the vector magnitude of the sample (i.e. image intensity) is represented by the colour of each glyph. If all the PCA is doing is accounting for the changes in image intensity then we would expect to see a very strong correlation between position in the visualisation and intensity. In all these plots it is clear that there is something more than just image intensity being presented.

In Figure 7.4(b) the data points and in particular the connectivity look very 'messy', for want of a better term. This 'mess' is a result of the way the manifold was constructed. The manifold is described as having 2-D connectivity, with

[2]Note that in these figures it is hard to perceive the depth of the image. The data for these figures, the OpenDX net used to generate them, and subsequent figures are included in the associated data distributed with this thesis.

one dimension being the subject, and the other being the pose of the subject. Clearly this is very arbitrary. There is no obvious natural ordering of the poses in question in the dataset and as a result it is extremely likely that the resulting manifold we construct is not continuous and differentiable. In this projection, we confirm what we would suspect — as discussed in Chapter 3, the structure of the parameter space intentionally bears little resemblance to the structures arising in image space. Nonetheless this dataset is still important to be included in this study; it will prove useful later on in demonstrating behaviour when the image manifold assumption we make breaks down. We can study, with this dataset, what happens when the basic assumptions (that there is only a small change between sequential images) about image manifolds do not hold. The fact that this dataset has also been successfully used with the Eigenfaces method still suggests that there is something important and interesting happening in image space. We can see from a number of these images how the principal components used allow us to distinguish at least one cluster, e.g. in the bottom centre most of the images of one subject are very tightly clustered in Figure 7.4(b). By projecting the faces onto an alternative three principle components we might well be able to distinguish between two classes which were indiscernible in the first three. By using more of the principle components returned, where each one added helps distinguish two or more classes we may well be able to, with the right choice of combination of principle components, distinguish all classes.

The results from the STRAIGHT_5 dataset, in Figure 7.4(c) are also interesting. For the most part the PCA projection has separated the data points out as we would hope, with temporally and spatially close points significantly closer in this space than more separated points. Furthermore we can note the effects of two facts we know about the dataset in images space. Firstly the obstacle that was present towards the middle of the path the robot took through the lab largely explains the 'hump' in the middle of the plot. Secondly the points where images

were not recorded due to lag/buffering issues in the acquisition system are very obvious. Additionally this plot seems to imply some inherent symmetry in the path taken by the robot, which makes sense in the context of the lab in which it was recorded (the path the robot took takes it away from a wall/desk area, into a space and past an obstacle before returning to another different wall/desk area).

The CHESSBOARD results, which are shown in Figure 7.4(d), also have a number of interesting features. We can see from the first principal component, shown in Figure 7.1(d) that there are patches where there are more white/black pixels, and patches where the image is more grey. The observed effect can in particular be explained when one considers that the images are in effect binary — a given pixel is always either black or white. This fact, combined with the discrete sampling of the CHESSBOARD results in a situation where there are only a small, finite number of ways to align a chessboard pattern with the eigenvector, owing to the fact that in a dataset with 10,000 images there are really only 100 unique images. The 10px $\times$ 10px square pattern 'wraps' around, causing this repetition. As a result small changes will result in a small shift in the image space, but the minimum size of a change is constrained. This means that at each point we see a number of samples, where the repeating pattern has overlaid them, as well as a system in which the connectivity in parameter space corresponds well to the neighbours in this projected subspace.

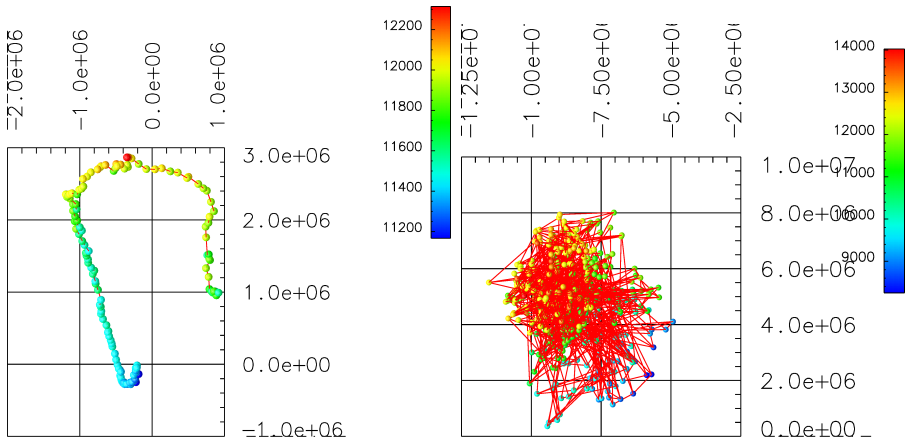
The BRUSH dataset results in Figure 7.4(a) shows almost exactly what one would expect to see given what we know about how the manifold was sampled. Through this visualisation we can see visual evidence here of some structure in image space. The point at which the least amount of surface area of the brush itself is fairly self-evident, being the sharp point of inflection, around the highest value point in the visualisation. This point is the brightest because the background is significantly brighter than the brush itself, and this orientation is such that the maximum background possible is visible. It is also the sharpest curved point of

the manifold is expected because the geometry of the situation means that on either side of this point the amount by which the pixels vary from image to image will be greatest, because more of the brush will appear/disappear between images. This connection between the content of the image and the behaviour observed in image space was already discussed in the previous chapter.

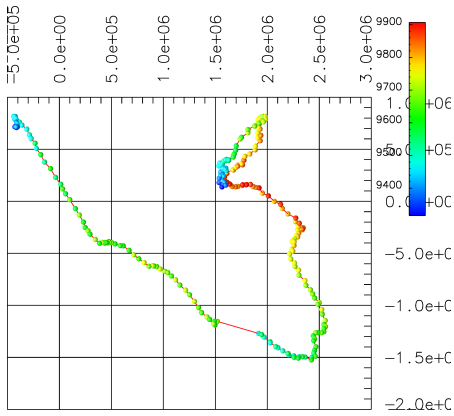
The interesting observation to take from all of these visualisations, except for the FACES which we discussed previously, is that the structures we see in the figures are all relatively simple. Connected images are close to each other and in these subspaces at least, the curves or surfaces shown seem to be relatively smooth. This is an important, positive observation, which we will build upon further in Part III by attempting to construct geometric models of these structures.

When the method is changed to the 'mtfast' method the results now change too; this is illustrated in Figure 7.5. Notice that here the CHESSBOARD is missing, owing to the fact that, as we mentioned previously, the 'mtfast' method is actually somewhat slower than the regular method when the number of samples is high and the resolution is low. In general here though the plots seem comparable to those of the previous method, Figure 7.4. The biggest visible change appears to be in the plot of the STRAIGHT_5 dataset. Here the plot seems to curve far more than previously, which is arguably less reflective of the path of the robot than previously. Other than this though the plots offered seem to be less useful for discerning the shape of the underlying manifolds. This is to be expected however — in the general case there is no fundamental reason to expect that the shape of path of the camera in world space would have much direct influence on the shape of the path of the camera in image space. Rather interestingly, although it is unclear why, the STRAIGHT_5 plot includes a very linear stretch now at one end of the dataset. On the other hand the basic topology (e.g. closed loop) of the path of the camera is reflected still in these visualisations.

From this, and the additional results not shown here, but included in the



(a) Brush dataset (b) Faces dataset



(c) Straight_5 dataset

Figure 7.5: PCA plots of the datasets, using the 'mtfast' method

appendix we can (informally) conclude that in general the results of the 'mtfast' method, whilst potentially useful, are inferior to the 'impca' method. We will therefore default to preferring the other method, except where it is not feasible. In view of this for the remainder of the chapter we will omit the figures from the 'mtfast' method, with just one interesting exception.

7.3.2 Strategies for sub-space selection

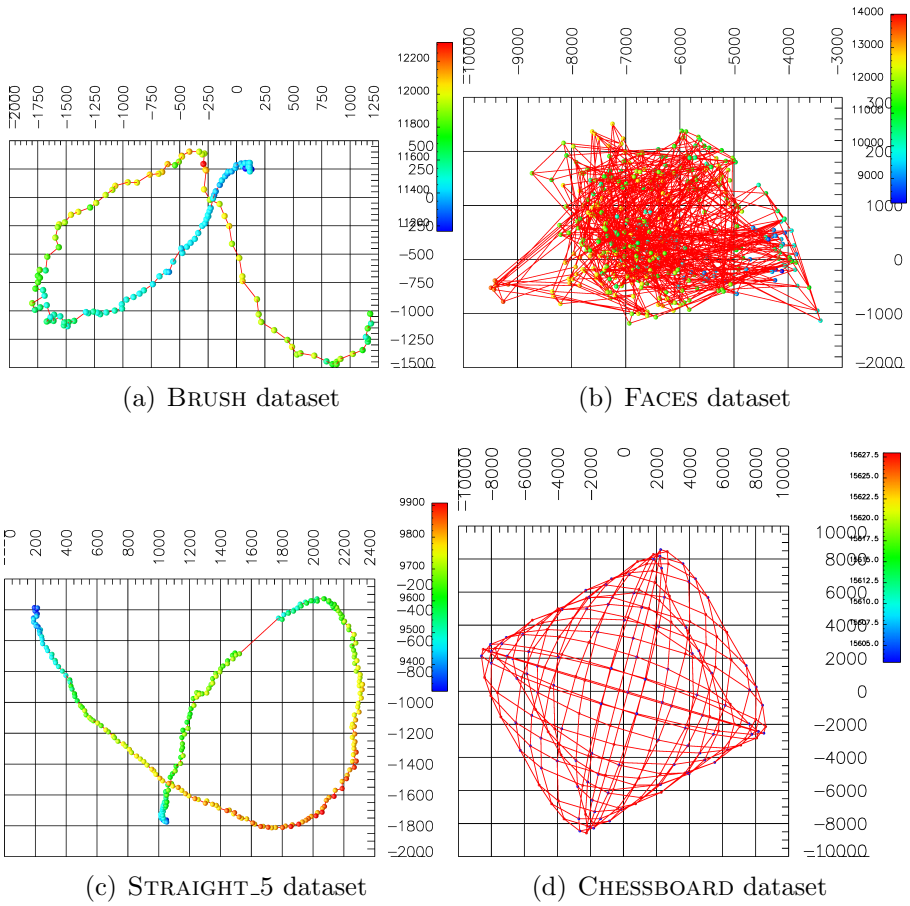
The question that arises naturally is what the remainder of the eigenvectors look like and further, how useful they would be in producing improved visualisations of the datasets. What we really seek from a visualisation is a clear separation of the datapoints, in a non-linear, or multi-dimensional way, which exhibits some

features that would appear to correlate with what we already know about the image manifolds themselves. In attempting to address this we include at this point a number of alternative selection strategies:

- eigenvectors associated with the 2^{nd}, 3^{rd} and 4^{th} largest eigenvalues (Figure 7.6),
- eigenvectors of the smallest three eigenvalues (Figures 7.7 and 7.8),
- eigenvectors of the smallest non-zero (to the working precision of the hardware and algorithm) eigenvalues (Figure 7.9),
- the eigenvector corresponding to the median eigenvalue and its two immediate neighbours (Figure 7.10).

This is clearly not an exhaustive list of all possible selection choices. It is in fact a very limited set of choices, but it does however include choices from the beginning, middle and end of the range of eigenvalues. The minor components (with the smallest eigenvalues) are unlikely to be particularly useful. We include them here out of completeness. The combinatorics of the setup mean that even with very small datasets it would not be feasible to consider every single combination (or even permutation) of eigenvectors for use as a subspace.

We can see from Figure 7.6 that plots using eigenvector numbers 2, 3 and 4, for the most part at least, are very similar to just using the first three eigenvectors, shown earlier in Figure 7.4. The overall structures shown here do not radically differ from the previous ones. This is not really too surprising given that two out of three of the eigenvectors in question are still the same. We could have chosen to slide this 'window' further down the list of eigenvectors, but as we shall see the later eigenvectors are less useful, and we know from the literature that discarding the first eigenvector is a reasonable choice (Jenkins, 2003; Poland and Zeugmann, 2006). The FACES manifold shows least change. It was 'messy' and is still 'messy',



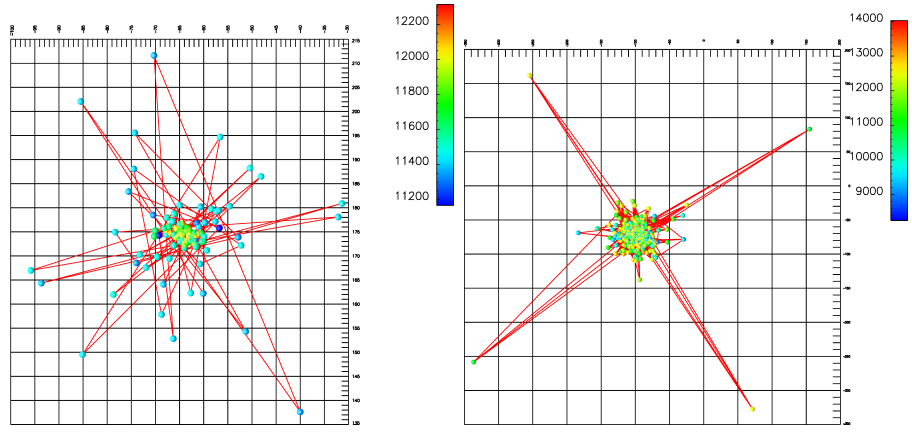
(a) Brush dataset (b) Faces dataset

(c) Straight_5 dataset (d) Chessboard dataset

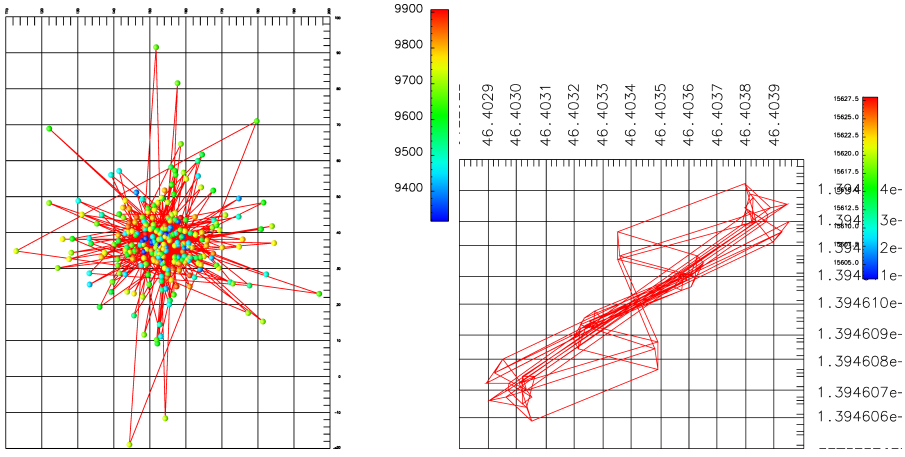
Figure 7.6: PCA plots of some of the datasets, 'first but one' eigenvectors

and although that mess may now be different it is no more or less useful to us as a visualisation of the underlying features in image space. It is interesting, and potentially useful however to observe that the Brush dataset now curves much more smoothly around the point of inflection (identifiable as the highest valued point in both) where the brush flips visible sides. The Chessboard visualisation is slightly different — we are using a different eigenvector after all — but still exhibits the same features for the same reasons as were outlined previously. In general from these results and the rest of the datasets which are only shown in Appendix D it seems that 'first but one' is as useful as the first three, and complementary to it.

The smallest eigenvectors, as illustrated in Figure 7.7 and further in Appendix D, represent a very poor choice for a visualisation. With the exception of the Chessboard dataset none of the high-level structures that would be useful

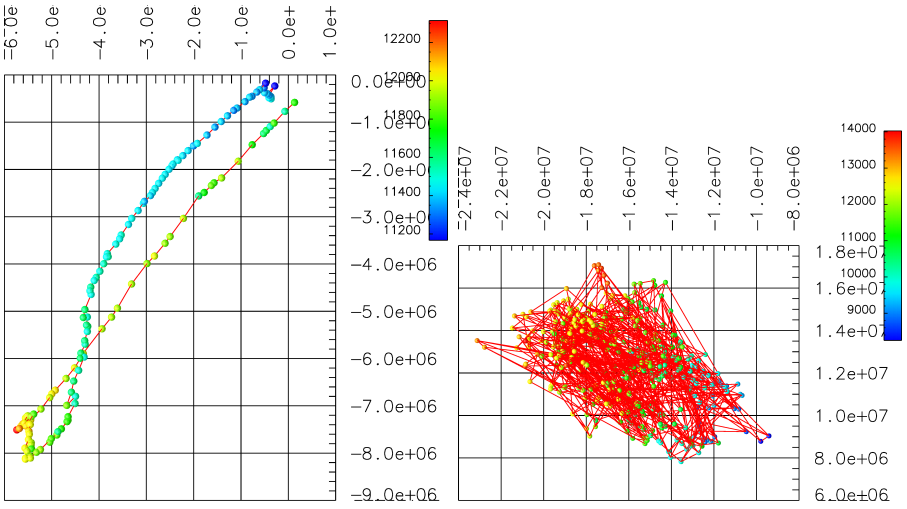


(a) Brush dataset (b) Faces dataset

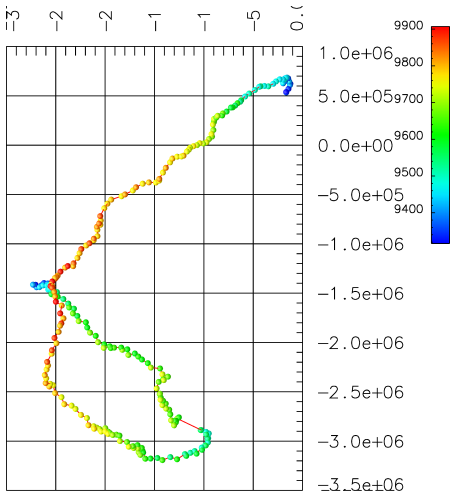


(c) Straight_5 dataset (d) Chessboard dataset

Figure 7.7: PCA plots of some of the datasets, smallest eigenvectors, using the 'impca' method



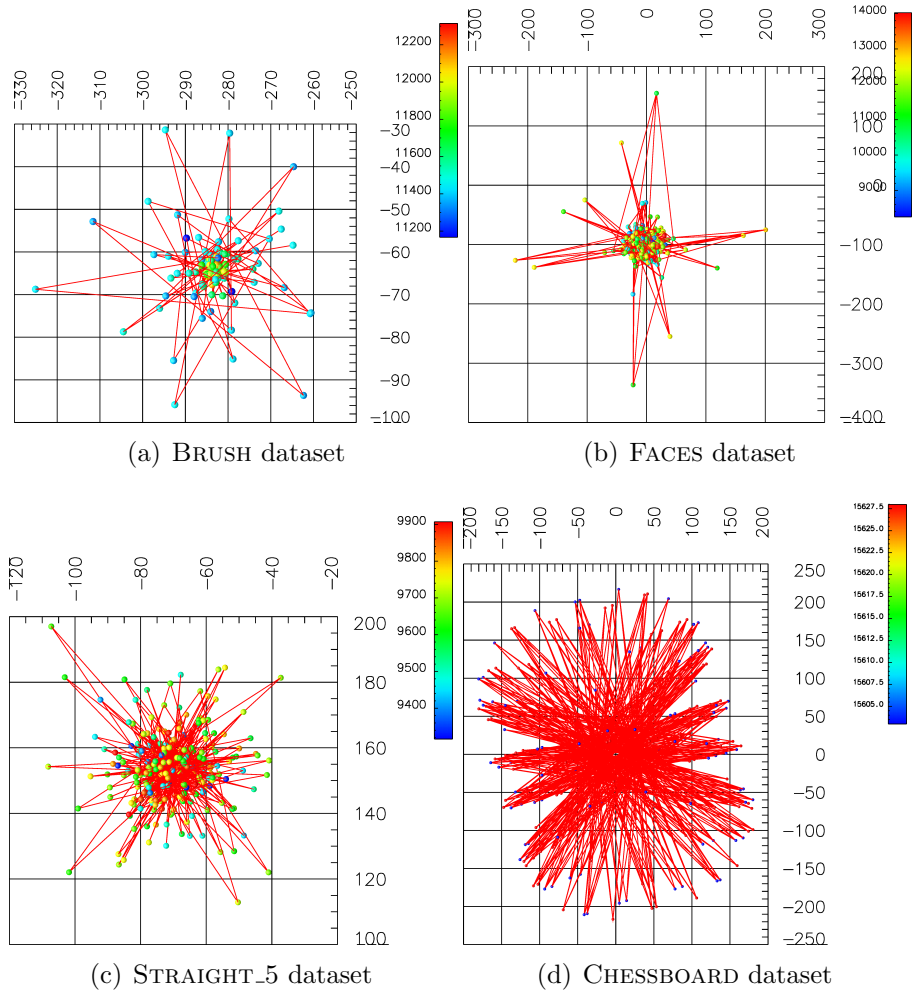
(a) Brush dataset (b) Faces dataset



(c) Straight_5 dataset

Figure 7.8: PCA plots of some of the datasets, using the 'mtfast' method, smallest eigenvectors

to see in a visualisation are clearly present, and all of them just succeed splitting a few, apparently random images out from the dataset. In practice these are likely to be noisy points, although they are of little interest to us here. The Chessboard visualisation now is the clearest of the visualisations, with still some indication of the underlying translation in this image, which was previously much more visible. It is safe to disregard the smallest eigenvectors as being useful for our visualisations.

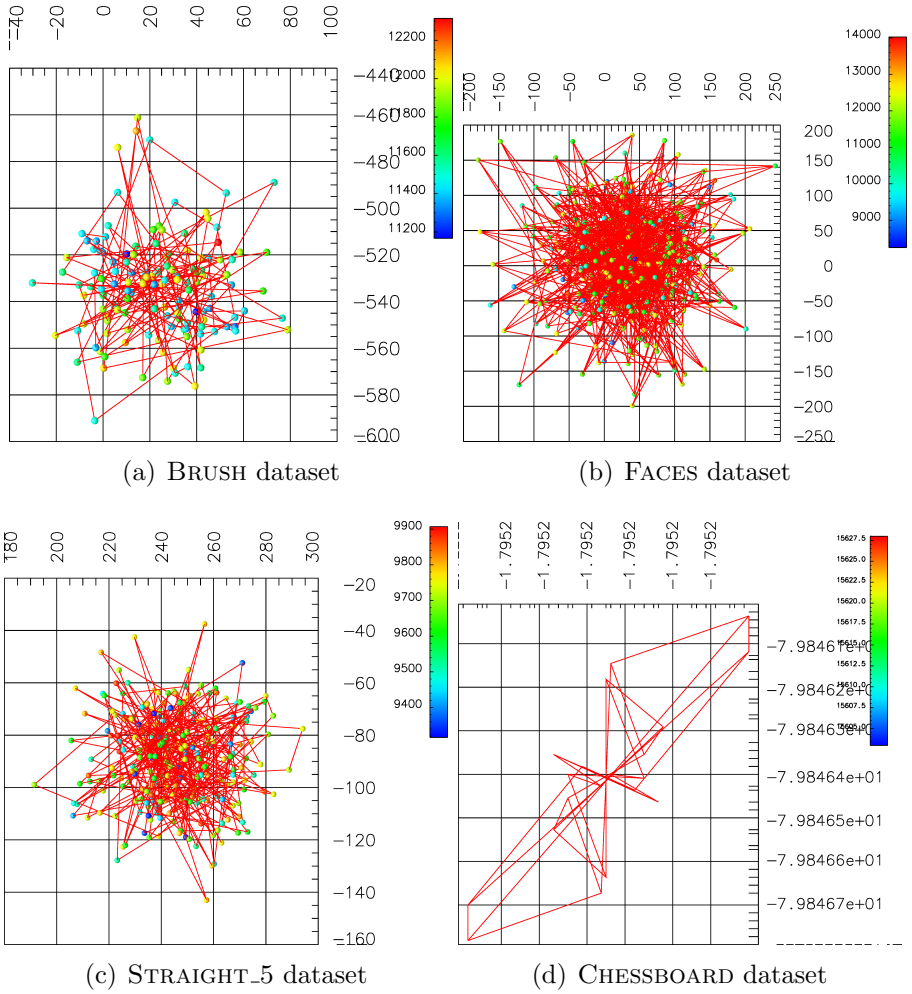


(a) Brush dataset (b) Faces dataset

(c) Straight_5 dataset (d) Chessboard dataset

Figure 7.9: PCA plots of the datasets, smallest, non-zero eigenvectors.

One interesting further point arises however if we take a moment to consider the smallest eigenvectors, as returned by the 'mtfast' method. These are illustrated in Figure 7.8. In this case it seems that we do still see some of the higher level features of the image manifolds. The explanation for this can be found by inspection of the eigenvalues produced by the two methods on the same dataset. In the case of the 'mtfast' method the eigenvalues cover a smaller spread (see Appendix D), with generally a lower maximum and a higher minimum. As a result of this the first few eigenvectors do not explain the dataset as well in the 'mtfast' method, leaving more of the dataset to be explained by the less significant eigenvectors, another reason for not choosing 'mtfast'.



(a) Brush dataset (b) Faces dataset

(c) Straight_5 dataset (d) Chessboard dataset

Figure 7.10: PCA plots of the datasets, median ±1 eigenvectors.

From the inspection of the resultant eigenvalues we notice that for the regular (i.e. not 'mtfast') method there is usually a large jump from 10^4 straight down to 10^{-6}. As a result we consider a fourth selection method: what happens when we consider only the eigenvectors with eigenvalues greater than 1? In practical terms this means that the value 1 makes a very robust threshold for detecting 'non-zero' eigenvalues, given that there is usually a minimum of an order of magnitude between this and any other eigenvalues. The results shown in Figure 7.9 are generated using this threshold. Once again however, we conclude that the associated eigenvectors do not provide a useful basis for a visualisation of our datasets and therefore will not consider it further as a sensible eigenvector choice.

Finally we consider the eigenvectors towards the centre of the range of eigenvalues. In Figure 7.10 we find the median eigenvalue, and use the corresponding eigenvector, as well as that of its two neighbours as the basis of our visualisations. Yet again we are forced to conclude that this does not represent an improvement over any of the previous eigenvector selections we have considered. This is also seen in the figures in Appendix D.

Upon review this section seems to confirm our intuition that the first few eigenvectors make the best basis for visualisations. It seems that in a number of cases excluding the first principal component in favour of the subsequent ones can improve the visualisations, with little or no discernible detrimental effect in cases where it does not improve it. As a result of this for the remainder of this thesis when we produce PCA plots we will, unless otherwise stated, default to selecting the eigenvectors corresponding to 2^{nd}, 3^{rd} and 4^{th} largest eigenvalues.

7.3.3 Alternative strategies

Here we have chosen to use PCA to visualise our datasets (and later results), which is linear and compared to some techniques (Tino et al., 2002) quite simplistic. This linearity does have a number of advantages for our work however. Firstly we are interested in the curvature of our models compared to the real data. Using a linear projection allows us to directly observe some of what is occurring in image space, by reducing the dimensionality sufficiently so as to permit visualisation. Secondly, the availability of robust, ready to use implementations of the algorithms required to implement draws us towards this.

In future work, especially if topological properties become more important, alternative visualisation strategies will become increasingly important.

7.4 Summary

In this chapter we have seen:

- how PCA can lead to principled visualisation of image manifolds;
- two useful algorithms for performing PCA experimentally on image manifolds;
- the results of performing PCA on some of our image manifolds;
- some visual evidence from the PCA projections to support the image manifold hypothesis;
- several approaches to selecting which eigenvectors to use for the re-projection;
- that two of these approaches are more useful than the rest, and selected one of these approaches (namely 'first but one') for use throughout the remainder of this thesis, on the basis of our observations applying it to our datasets;
- a brief discussion of these results, where we can clearly see evidence of simple structures in the resulting visualisations, and the implications of this for our work.

Full details of the results of performing PCA on all of the datasets were omitted from this chapter in the interest of brevity. They are however included in Appendix D.

In the next chapter we will explore:

- a baseline method for modelling the structures we have been investigating in this chapter;
- a visualisation of the output of our model that uses PCA and is directly comparable to the results presented here;
- results from the above.

Part III

Modelling image manifolds

Introduction

Previously we have looked at the concept of an image manifold, and seen how under certain conditions sequences of related images that vary between them by only small amounts in some parameter space can form manifold-like structures. The question which naturally poses itself after this discovery is "can we model these manifolds to capture their appearance?". This part is devoted to answering this question. To address this question we look primarily at two different generic surface modelling techniques that have been proposed in the literature as solutions for use in the typical CAD graphics domain.

The two methods we study are NURBS and the PDE surface method. These methods were selected over other possible alternatives because in the 3-D world (e.g. CAD/CAM, games) they offer a very generic approach to modelling curves and surfaces. The PDE surface method is a more recent development in this area and offers and interesting alternative to NURBS. Since the aim of our research (Chapter 4) is to construct as generic a representation of the manifolds as possible these two approaches, which are feasible to generalise to higher dimensions (intrinsic and extrinsic) are a natural choice. Of course, by choosing to take this approach we accept the inevitable costs introduced by working in the full image space. The intention is that by taking such a generic approach we can offer an approach to modelling image manifolds (and hence IBR and appearance based vision problems) which is more generic and broadly applicable than any of the existing literature (See Chapter 3, Section 3.5 for a discussion on this).

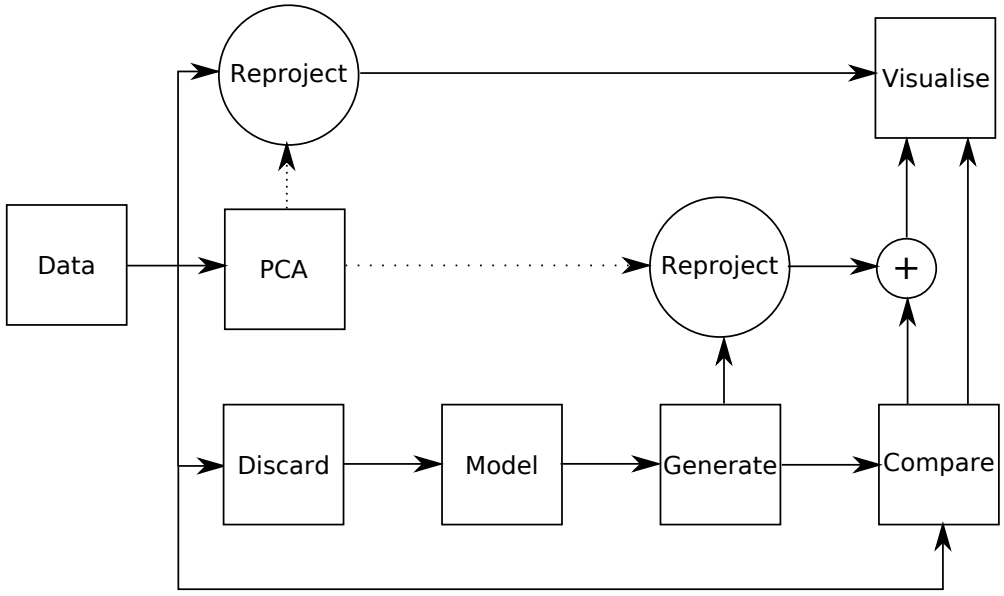
The two proposed modelling techniques are discussed seperately in Chapters 9 and 10 respectively. Prior to this however we discuss, in Chapter 8, a naïve method as a starting point.

Experimental process

The experimental process we use is summarised in the figure below. In Chapter 7 we have already seen one path of process when we performed PCA on our datasets and used this to generate a suitable subspace in which to re-project the original data and visualise it. Following on from this we use a similar approach to visualise the output from many of our experiments. As previously discussed (see Chapter 5) the core methodology we use when experimentally testing the models of image manifolds we use can be summarised as:

1. Data
2. Discard some input images
3. Construct model
4. Re-generate discarded images from the model
5. Evaluate generated images

This corresponds to the "discard, model, generate, compare" element of our process diagram, shown below:



Schematic of data flow and processes used in experiments

Throughout the next three chapters we introduce three models for image manifolds. Each of these are tested under different configurations through this one process. The 'compare' procedure is our evaluation, and uses the five image metrics we introduced in Chapter 5 to compare the *synthesised* ('generated') images, which represent the discarded images, with the known, actual sampled images which are our ground truth. In most of the subsequent visualisations we use the subspace identified by having performed PCA to show a re-projection of the *synthesised* images, with the position being determined by the synthesised image and the value (i.e. colour) of the glyph determined by the results of the comparison with the original image; this combination of the re-projection and the result of the comparison is symbolised by the '+' in the figure above. In a number of instances it is more helpful to view the results of the comparison in the parameter (i.e. intrinsic) space of the manifold in question, hence the alternative, direct visualisation of the comparison in this diagram. This will be identified where appropriate, however it is easily distinguishable by the uniform gridded positioning of samples in the space.

Our models are designed such that they represent not only the position in image space (i.e. the appearance of the images), but that they also faithfully model the original parametrisation of the dataset. This means that when we evaluate our synthesised images we deliberately, explicitly measure the distance between the ground truth and the synthesised image and not the distance between the synthesised image and the closest manifold point. Thus if our model has correctly modelled both the parametrisation and the positions this distance will be low and should one or both of those be incorrect then a larger distance will be recorded by the metric. Note that this is similar to the evaluation method proposed in (Souvenir et al., 2006), although we consider a wider selection of metrics.

Finally, the procedure by which we discard images is fixed throughout this thesis, with the exception of Chapter 12, where we introduce and discuss several alternative strategies for selecting which images to discard.

Throughout the whole of this part of the thesis we include only a brief summary of the results in each chapter, using 'stand-alone' discussions and identifying important parts of our results. This makes it possible to present a fuller discussion and analysis in Chapter 11. Deferring discussion in this way enables us to present our discussion once all of the details and results of each method have been dealt with, in order to draw broader conclusions.

Chapter 8

Linear combinations

8.1 Introduction

Prior to our study of NURBS and PDE surfaces we introduce *piecewise linear combinations* of nearest neighbours (in parameter space). This is provided as a baseline for comparison and is intended to be a simplistic method for modelling image manifolds that does not in anyway seek to preserve any higher order continuity. It still does serve as sensible starting point, and a good comparison to higher order methods.

The basic premise of the method of linear combinations, illustrated in Figure 8.1 is that an image with a given parameter u is computed from the two closest neighbours, with a linear weighting given to each neighbour. In Figure 8.1 $u = 0$ and $u = 1$ are given as sample points, meanwhile the new point $u = 0.5$ is to be computed. The hollow circle illustrates where this simple scheme would place an intermediate point, but this can only be correct if the true path taken is both a straight line (the piecewise linear combinations can only ever produce points that fall on the straight line between the two points), and traversed at constant speed (otherwise halfway along will not be a geometric midpoint). Often this means that the line between the real point and the calculated point will not be perpendicular to the line between the two samples used to generate it. The disparity between

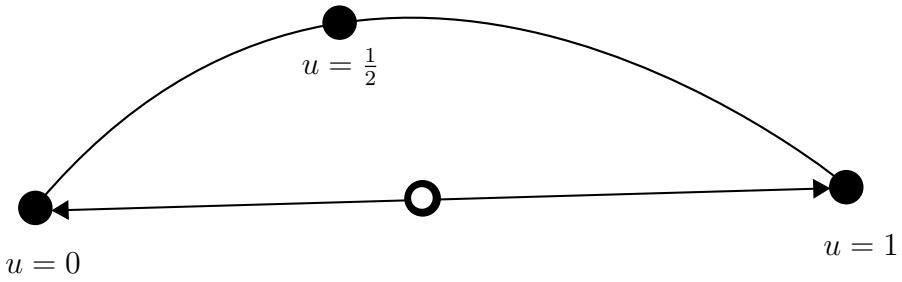


Figure 8.1: The image at $u = 0.5$ is computed using the samples at $u = 0$ and $u = 1$

Figure 8.2: This image, which clearly shows 'ghosting' is generated as a half-way image between two real photographs

our computed position and the indicated real position can potentially be large.

If we further consider the above example, but instead consider points in image space we can see overwhelming (but anecdotal nonetheless) evidence for the non-linearity of most 'real-world' image manifolds. This is of course exactly what we saw in Chapter 6 with our midpoint distance measurement and again in Chapter 7 in almost all of the PCA plots. Figure 8.2 illustrates it for one such example. We will try to build models in the next two chapters that better approximate the mid point between the two sampled images, but for now we will implement and investigate this simple linear method.

8.2 Implementation

We have sample points, corresponding to images, $\underline{Q}(x_1, x_2 \dots x_{d-1}, x_d)$, which form a d-dimensional sampling grid, where the x_i are the integer coordinates on the grid, in the interval $[0 \dots N_i]$, where the N_i's define the size of the grid. Using this we seek to define a continuous function $\underline{S}(u_1, u_2 \dots u_{d-1}, u_d)$ which defines any point on this grid.

This function $\underline{S}$ defines any point within the grid as a piecewise linear interpolation on a hypercube of dimension d, defined by the 2^d points on the grid $\underline{Q}$ which completely enclose the point we are calculating. We find these 2^d points, and label them as $\underline{V}_j$ by finding the coordinates on the grid of the "lower left corner" of the hypercube. Thus the coordinates of this lower left corner $\underline{V}$ are given from the u_i:

$$\underline{V}_0 = \underline{Q}(\lfloor u_1 \rfloor, \lfloor u_2 \rfloor \dots \lfloor u_{d-1} \rfloor, \lfloor u_d \rfloor), \tag{8.1}$$

and the remaining $\underline{V}_j$'s are selected by replacing selected $\lfloor u_i \rfloor$'s with $\lfloor u_i \rfloor + 1$ in every possible permutation. This gives us our 2^d $\underline{V}_j$'s, where the index j, when viewed as a binary number, indicates the offset from the lower left corner of the hypercube, e.g. in 3-D 010 ($j = 2$) would represent $\lfloor u_1 \rfloor, \lfloor u_2 \rfloor + 1, \lfloor u_3 \rfloor$ while 110 ($j = 6$) would represent $\lfloor u_1 \rfloor + 1, \lfloor u_2 \rfloor + 1, \lfloor u_3 \rfloor$. This approach is similar to the system used in (Lorensen and Cline, 1987) for the marching cubes algorithm and illustrated in Figure 8.3.

Next, in order to simplify our notation further we apply a translation to the entire hypercube, such that the origin is at 0, and the u_i fall in the interval $[0, 1)$. We refer to the transformed u_i as:

$$(u'_1, u'_2 \dots u'_{d-1}, u'_d) = (u_1 - \lfloor u_1 \rfloor, u_2 - \lfloor u_2 \rfloor \dots u_{d-1} - \lfloor u_{d-1} \rfloor, u_d - \lfloor u_d \rfloor). \tag{8.2}$$

From the definition of our grid we can indeed verify that this will form a unit

hypercube as the grid coordinates were defined to be integers across a given range, starting at 0. In the general case however there may be grids which do not meet this requirement, for which the transformation required for the u_i's may be more complicated than just a single translation.

From this we can define our interpolation:

$$\underline{S}(u_1, u_2 \dots u_{d-1}, u_d) = \sum_{j=0}^{2^d} \underline{V}_j C(1, j, u_1') C(2, j, u_2') \dots C(d-1, j, u_{d-1}') C(d, j, u_d'), \tag{8.3}$$

where

$$C(n, m, u_i') = \begin{cases} u_i' & \text{if bit } n \text{ in } m \text{ is set} \\ (1 - u_i') & \text{otherwise} \end{cases} \tag{8.4}$$

This is perhaps best clarified through an example, given in Figure 8.3. In this instance the 3-d interpolation is given as follows:

$$\begin{aligned} \underline{S}(x, y, z) \quad = \quad & \underline{V}_{000}(1-x)(1-y)(1-z) + \\ & \underline{V}_{001}(1-x)(1-y)z + \\ & \underline{V}_{010}(1-x)y(1-z) + \\ & \underline{V}_{011}(1-x)yz + \\ & \underline{V}_{100}x(1-y)(1-z) + \\ & \underline{V}_{101}x(1-y)z + \\ & \underline{V}_{110}xy(1-z) + \\ & \underline{V}_{111}xyz, \end{aligned}$$

where x, y, and z are u_1', u_2' and u_3' respectively.

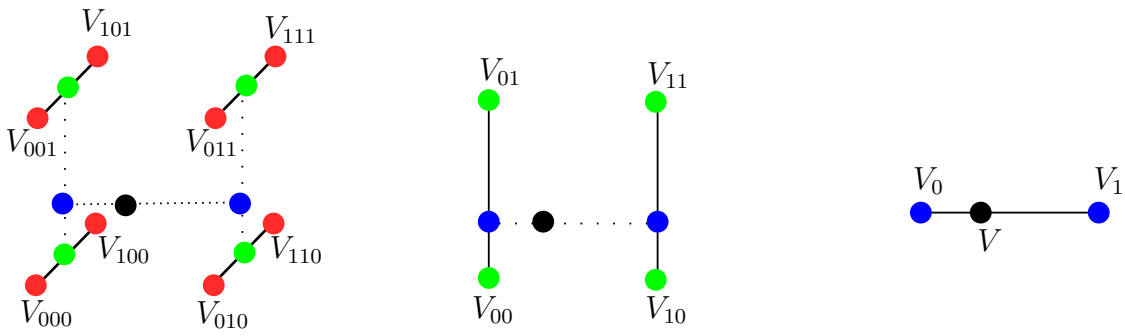


Figure 8.3: Recursive N-dimensional implementation of linear interpolation

Correspondingly the 2-d interpolation is given by:

$$\begin{aligned} \underline{S}(x,y) &= \underline{V}_{00}(1-x)(1-y) + \\ &\quad \underline{V}_{01}(1-x)y + \\ &\quad \underline{V}_{10}x(1-y) + \\ &\quad \underline{V}_{11}xy, \end{aligned}$$

and the 1-d interpolation is simply:

$$\begin{aligned} \underline{S}(x) &= \underline{V}_0(1-x) + \\ &\quad \underline{V}_1 x. \end{aligned}$$

In practice there is a more efficient algorithm for computing this, which is outlined in (Rovatti et al., 1998).

8.3 Experimental setup

The experimental setup for the linear model is relatively simple. The only parameters we have to vary between experiments is the scale at which we build the model, and the dataset itself. We refer to the scale as the discard gap, i.e. the number of input images we discard between the samples we retain.

In the specific instance we use the following set of discard gaps (see Section 5.2,

page 97), $g = 2$, $g = 5$, $g = 10$, $g = 20$, $g = 25$, $g = 50$, $g = 100$. This setup is described in more detail earlier, in Chapter 5. We will also use the same set of configurations as the basis of our NURBS and PDE model configurations in the proceeding two chapters, which ensures results are directly comparable.

8.4 Results

We run our linear model with each of the aforementioned discard gaps, for every one of our datasets. Furthermore we use each of our five metrics to evaluate the results. This means that after running all of our (linear model) experiments we potentially have 980 visualisations of the output of our models! We have therefore produced an 'at a glance' summary table, which we refer to as the basis of our discussions of the results.

In Appendix E.1 we have included summary tables for all of the metrics we discussed in Chapter 5. During the discussion we focus mainly on the taxi and pdiff metrics. These two are by and large representative of the sorts of results we see from the other metrics.

In a number of instances the experiments failed to finish, or were not ever scheduled to run. In the context of the linear model this is usually due to an insufficient number of samples for the gap width, (i.e. larger gap than our total number of image). In a few instances the metric itself failed (e.g. SIFT returns no key features at all for one of the images, which means that it is not possible to (meaningfully) measure the cost of a mapping between the two images, see Chapter 5). Additionally a number of results are currently missing because the PCA failed to complete (normally exceeded storage or time limits), and due to the way the experiments were run, PCA was considered to be a prerequisite for this stage of experimentation.

There is one further important consideration which the reader should be aware of, when interpreting the numerical results; the results for any given dataset with

any given value of g are not directly comparable to the corresponding results, with the same gap, for a different dataset. This arises from the fact that our sampling frequencies are not equivalent between any two given datasets. This is even true for repeat-runs of similar experiments, e.g. the STRAIGHT_N series, where factors like disk-buffer flushes and robot speed mean that sampling is neither uniform (in terms of space and time), nor identical between them. Furthermore the important consideration, as we will see over the remainder of this thesis, is not so much the spacing of samples in parameter space (which our g variable represents), but the spacing of the samples in image space. This of course isn't even uniform amongst all pairs of neighbouring samples of a given dataset, and hence not shown explicitly in our summary tables.

The mean (μ) and standard deviation (σ) values indicated in our results tables from this chapter and the remainder of this thesis are the mean error reported by the given metric for all of the images that were generated by running the model with the configuration in question. Similarly the standard deviation reported is the standard deviation of the errors reported by the metrics for all of the images produced. If all of the images reported an identical error it would be 0. It is not a reflection of the distribution of the error within the individual images in anyway. For instance it is not possible to discern from these tables if the error is the same across all pixels, or comes from only one pixel.

In the results we present here the metrics are not normalised, like they were for the visual navigation experiments in Section 5.1.5.1, because the choice of a value to normalise by is not so intuitive for the general case, and the normalisation itself offers little value. Where we visualise the results using PCA plots the colour of the glyph indicates the error reported by the metric for each given image, whilst the colour map shown along side indicates the range of errors present in the experiment.

8.5 Discussion

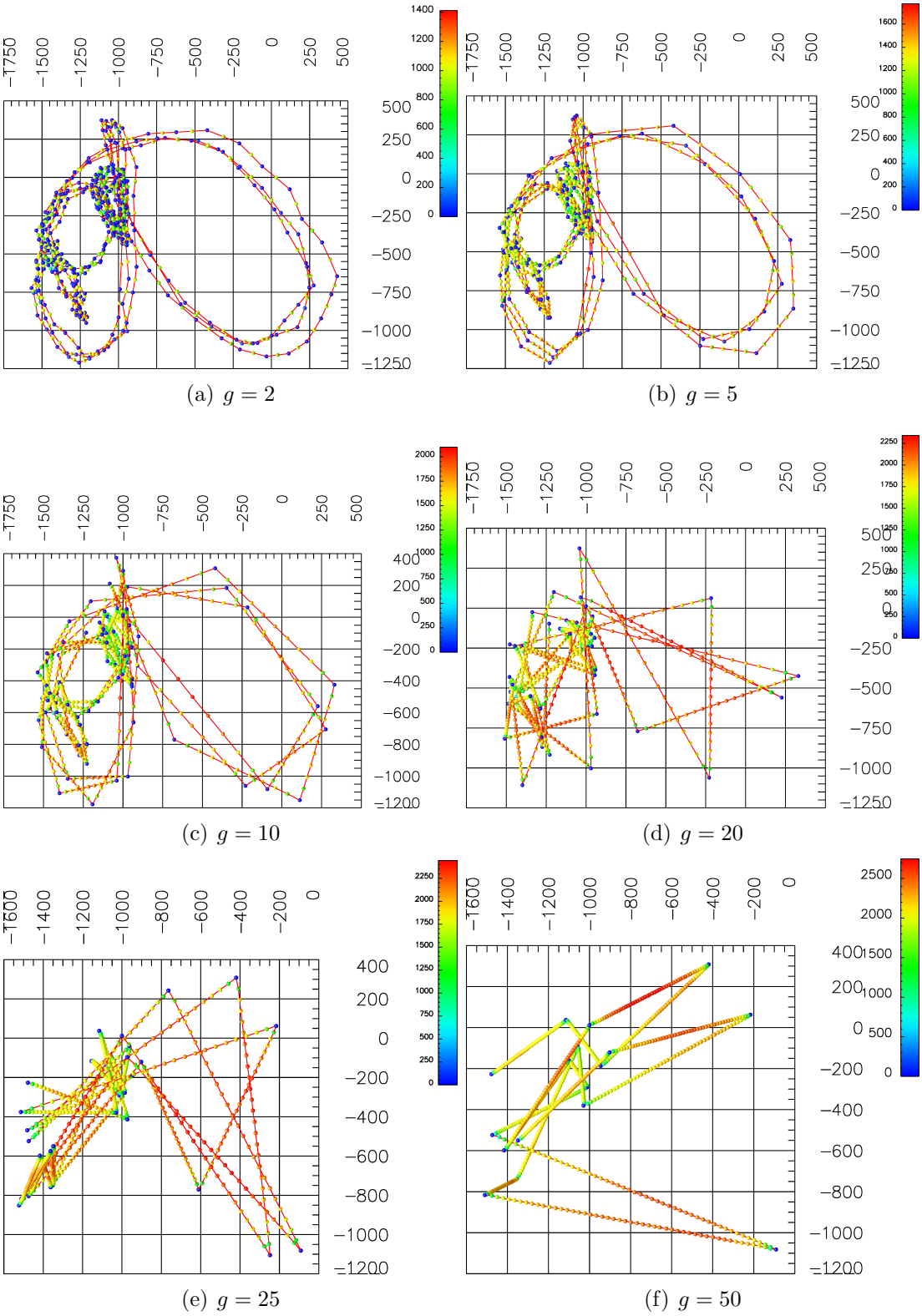
These results are only presented to provide a baseline for comparison against other methods. Therefore the discussion here is brief, and the results will be further discussed in the context of the other models we use, later in Chapter 11. Inspecting the detailed results (see for example Tables E.1, E.2, page 381), we can see that the error is not uniform, when we change either the gap, or the dataset itself. This is as we would expect given what we saw in Chapter 6 previously.

We will now look in more detail at a specific dataset, namely Circle_5, shown in Figure 8.4. This figure should be viewed in reference to the original PCA plot of the entire dataset, Figure D.8 (page 349). It is important to note here that what we observe in any PCA plot is only really representative of a small fraction of the whole image space. It is very hard to understand what the implications are for dimensions we cannot visualise. This is not a problem necessarily, provided we accept that some topological and other important structural features are likely to be misrepresented.

There are however, a number of observations to be made at this juncture. Firstly the three repeated loops the robot made in the lab are clearly visible from $g = 2$ through to $g = 10$. At $g = 20$ it is apparent that in this visualisation at least the structure is starting to be lost. That is not to say that it is not lost earlier in some of the other eigenvectors, or later in yet more. We can see, as we would expect that the error is not uniform in image space, or parameter space, which agrees with (and is exactly the same as) we saw in Chapter 6.

Predictably, in all cases the error increases as we increase the gap, until important features of the structure are completely absent. In Circle_5, by the time we reach $g = 50$ almost nothing of the structure is visible.

This would seem to imply that in practice, we might see the emergence of two (potentially) distinct thresholds from our metrics; firstly where the image fidelity itself degrades noticeably, to a point where an observer would notice; secondly



(a) $g = 2$ (b) $g = 5$

(c) $g = 10$ (d) $g = 20$

(e) $g = 25$ (f) $g = 50$

Figure 8.4: Visualisation of the detailed results for the CIRCLE_5 manifold

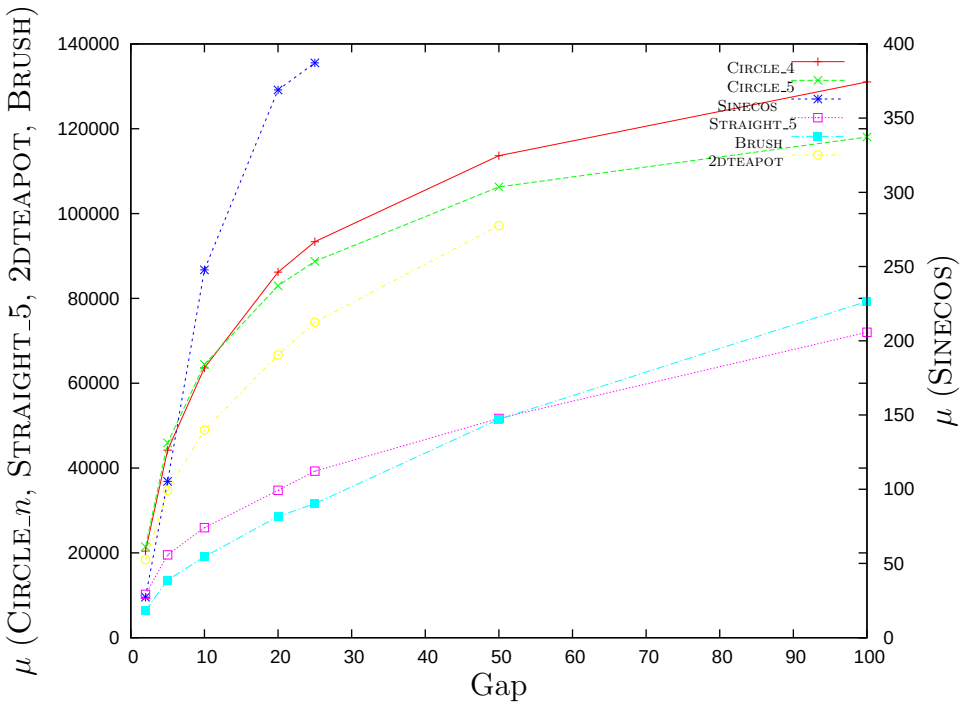


Figure 8.5: The mean error, vs gap for a selection of datasets

where the manifold structure/topology degrades noticeably. Whether or not these are two distinct thresholds is application dependent, because the maximum tolerable error is also application dependent.

At this point we could potentially measure experimentally, and derive theoretically, a value for the second of these limits, by measuring the frequency and using the Nyquist-Shannon sampling theorem. This direction however, would not really add much to this thesis. To calculate this however would require retention of far more sample images than desirable and as we shall see, we may be able to heuristically estimate this limit anyway. Furthermore we can safely assume, by definition, that this will always be larger than or equal to the 'visible loss of quality' threshold. This makes a good lower bound on the threshold. This is because exceeding the point where the whole structure of the manifold is lost through aliasing would have to have observably degraded consequences for visual fidelity.

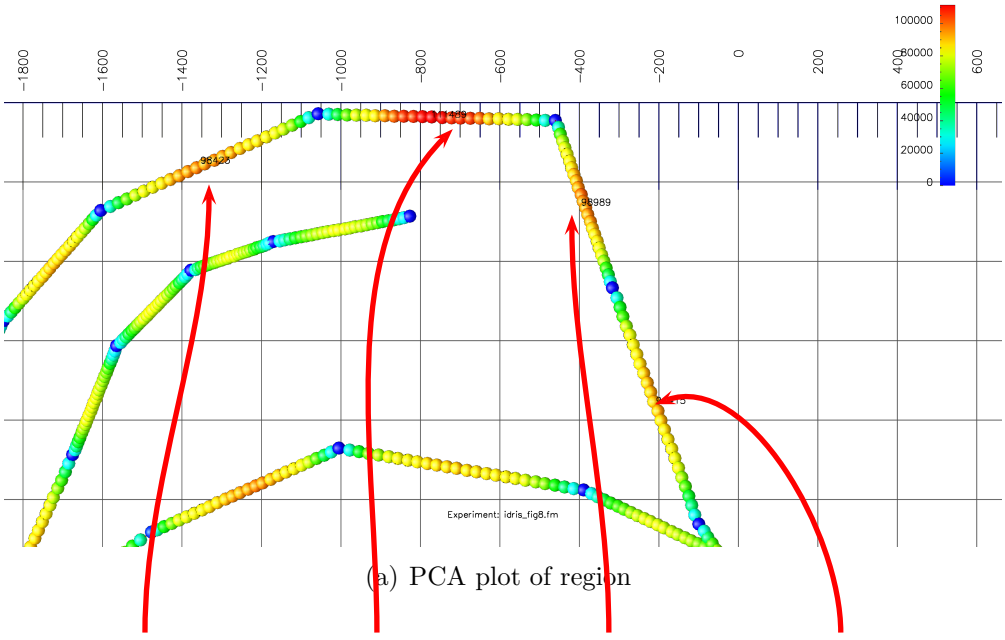
In Figure 8.5 we plot mean error against gap spacing. In this figure it seems clear that the mean error tends towards some limit; this also makes sense intuitively – the worst we could possibly do would be to generate an image on the

opposite side of the manifold from the samples used to generate it. This effectively means that the absolute maximum error we could ever see would be the same as the greatest distance between any pair of images (consider a scenario where the first n images fall on one point or cluster, and the next m form a straight line away from that single point, the next o images return to the cluster and the final p images are on the same point/cluster as the start). In practice though we don't get close to this maximum, as the circumstance in which it could occur are quite rare. A more realistic experimental limit might be half of the maximum observable distance. Furthermore the graph in Figure 8.5 plots *mean* error, not *max* error which further reduces what we observe, as compared to any theoretical limit.

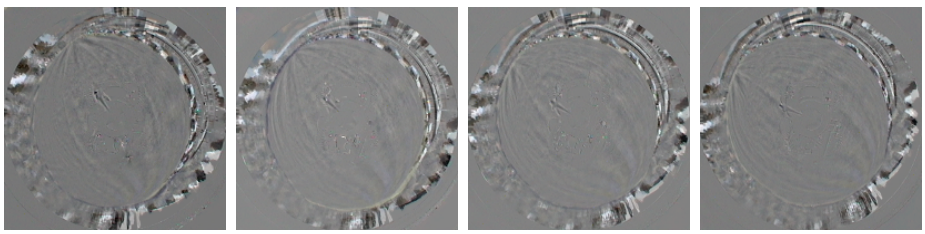
In many respects what we see here is just another take on the exact same measurements we discussed in Chapter 6. In this instance, however we have used a number of different metrics in our results, whereas previously we only considered the Euclidean distance, for purely geometric reasons. The presentation here is also the same as our presentation in the following two chapters, where we look at NURBS and PDE surfaces. We have therefore, crucially, established in this chapter a frame of reference, to which we can judge any future improvements we may make to our manifold model.

We can see from these results in general, that it is often true that the distance (at least in our PCA visualisations) is not always a function of the observed error. An example of this is illustrated in Figure 8.6, for the IDRIS_FIG8 dataset, with $g = 25$. In this figure[1] it is relatively clear that the second stretch of generated images (Figure 8.6(c), which is somewhat lighter in the difference image, indicating higher error than the rest) has a far higher error than the others, despite being the shortest of the four. That is to say that the sequence of images produced from the pair of samples with the least distance between them also has the highest error. This is important for a number of reasons. Primarily though it suggests

[1]Provided it is printed or displayed well, the difference in grey levels is quite subtle but visible on screen.



(a) PCA plot of region



(b) Error = 98423 Span len. = 2.39

(c) Error = 111489 Span len. = 2.32

(d) Error = 98989 Span len. = 2.53

(e) Error = 94415 Span len. = 2.38

Figure 8.6: An example of error (taxi) not being a function of the distance of the two samples used to generate it. (b)—(e) are difference images between the original, sampled image and the prediction by the model. Lighter areas of the difference images represent higher errors. Span lengths are measured using taxi and $\times 10^6$.

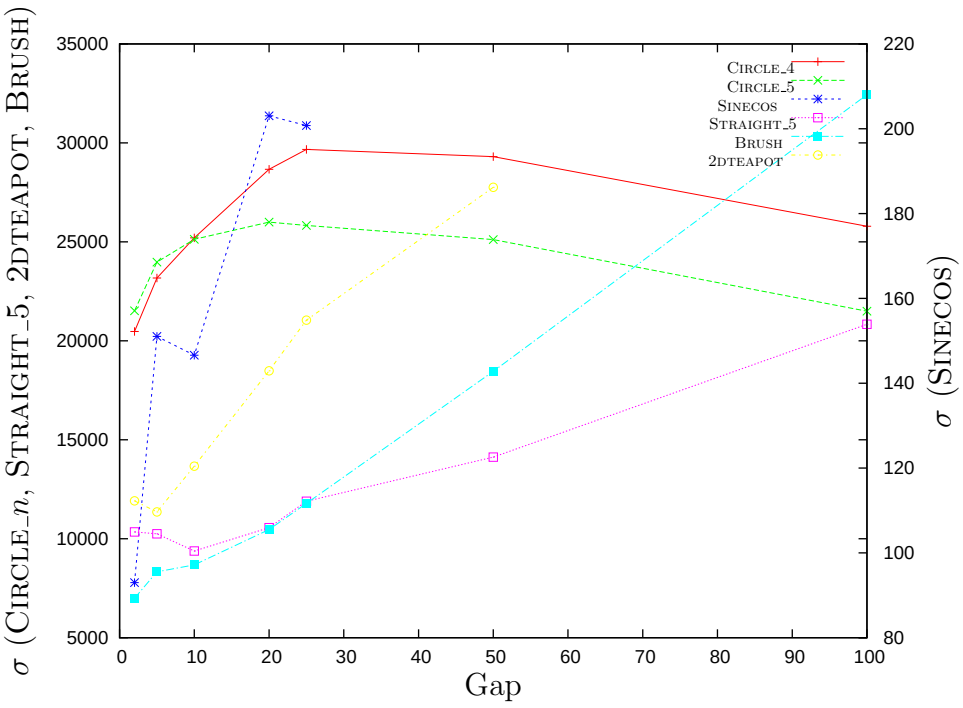
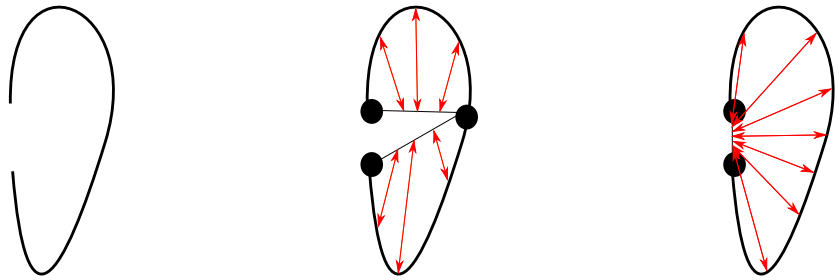


Figure 8.7: The standard deviation of mean error, vs gap for a selection of datasets

that our approach to modelling image manifolds is sensible — if the error were just a function of distance then there would be no courses of action available to improve the error, beyond just reducing the distance, yet this clearly shows it to not always the case. This shows that the curvature varies, which suggests there is scope for improving this result. We will explore this in the remaining chapters.

As we look at, and try to understand, the standard deviations of our errors (Figure 8.7), we can make several interesting observations. In many cases the standard deviation decreases as we approach the larger gaps. In the case of the pdiff metric this is trivial to explain; the pdiff metric merely counts the number of pixels with perceivable difference from the ground truth. As the gap gets larger, and we approach the bounds on the error we reach a point were almost all of the pixels are perceivably wrong. This means that more and more images will be returning the maximum observable distance, which in turn means that the difference between each measure will reduce until we reach the point where every image returns the maximum error and the difference between the errors is 0.

In the case of the taxi metric however there are more subtle reasons. We



(a) The path of the manifold.

(b) A possible sampling strategy that uses 3 sample points.

(c) Another possible sampling strategy, using just two points. The average measured error is larger, but the standard deviation is smaller

Figure 8.8: An hypothetical example of standard deviation 'peaks' as the gap increases. Measured errors are shown in red.

appear to have a serious aliasing problem here, which also relates back to what we mentioned previously about sampling frequency. Figure 8.8 illustrates one such scenario where this might occur. In this example, (c) will have a higher mean error than (b), since there are more places where the interpolation is further to the actual path taken. However (c) will have a lower standard deviation since the errors will be more similar.

The question then that we wish to answer is: 'is this an acceptable, compact representation of our image manifolds?' The answer to this is probably yes in many cases — with small gaps we see small errors, and closer inspection often reveals only small errors within these gaps. In some cases the illusion is actually quite convincing[2], even with a very crude model. An example of this is illustrated in Figure 8.9, where both pdiff and taxi report much lower errors than other stretches (Figure 8.10) of the same experiment. Furthermore if we considered an application, such as motion images, then the visible effect of the errors is further reduced[3]. This may also be acceptable in some scenarios, such as visual navigation, where even a (G^0) continuous approximation of a manifold would facilitate algorithm design. It would however be very foolish to stop at this point.

[2]There is little to no ghosting obvious in the generated images and visually smooth changes from image to image.

[3]Of course in practice this is much better served by video compression techniques!

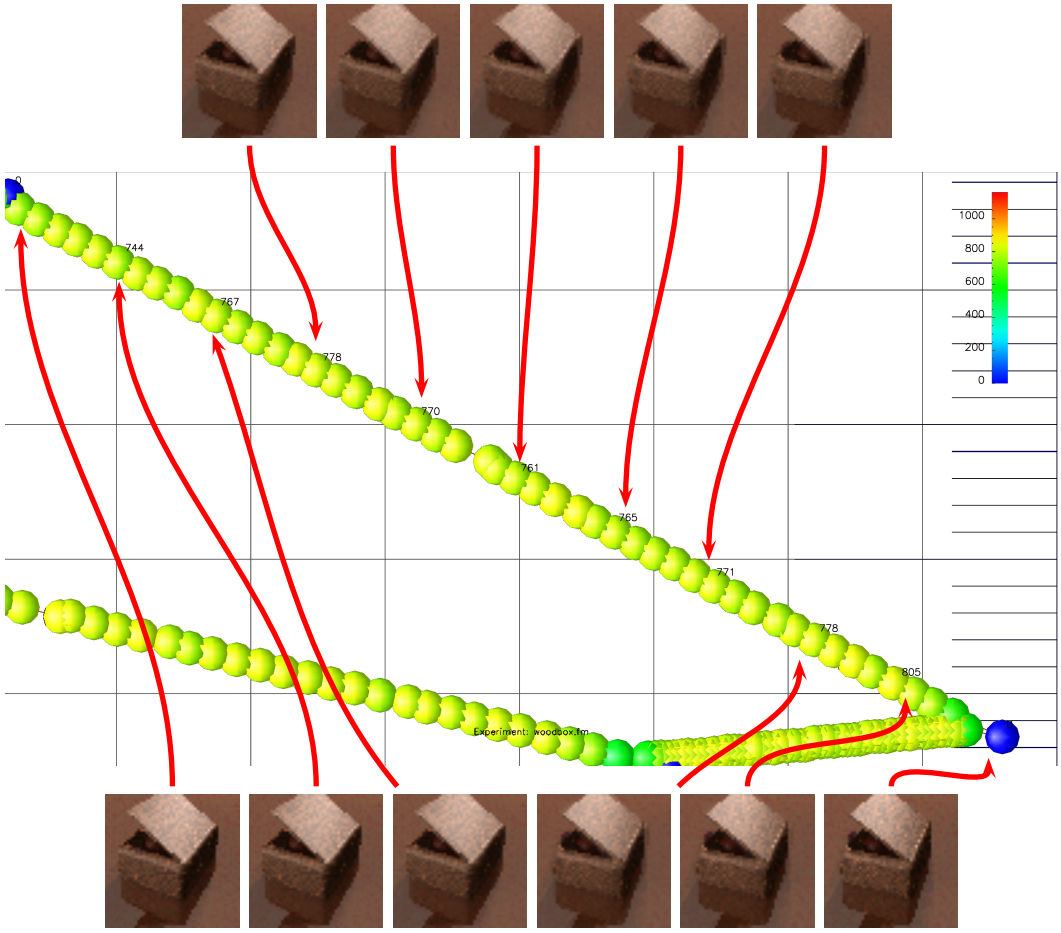


Figure 8.9: PCA plot, showing a stretch of the WOODBOX manifold in detail, $g = 50$, pdiff, linear model

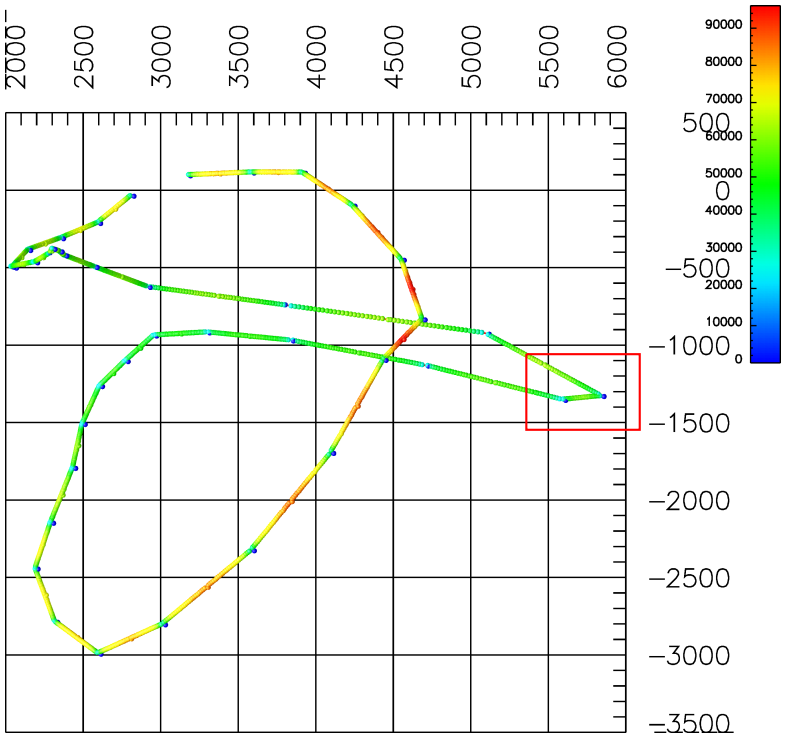


Figure 8.10: Overview of the whole WOODBOX manifold as shown in Figure 8.9. Detailed area is highlighted in red.

We know our image manifolds are generally highly curved, and we know that, crucially, our linear model completely ignores this. We therefore devote the next two chapters to consideration of the two alternative, schemes for modelling curves and curved surfaces, which we also propose for modelling image manifolds.

8.6 Summary

In this chapter we have seen:

- a simple model of image manifolds that uses piecewise linear combinations of sample points;
- how we can recursively formulate this to work for the general case;
- results of this applied to our selected image manifolds;
- a brief discussion of these results, and their usefulness as a basis for comparison to other models.

In the next chapter we will explore:

- improving our manifold model by extending and applying NURBS;
- an experimental setup for the NURBS model;
- results from the application of this.

Chapter 9

NURBS

9.1 Introduction

In the early 1960s and 1970s when CAD/CAM systems were in their infancy the automotive industry was searching for simple elegant mathematical descriptions of curves and surfaces. It was out of this need that Pierre Bézier, who at the time was working for Renault, developed the *Bézier curve*. It is from this that the Non Uniform Rational B-Spline (*NURBS*) curve developed. This historical perspective provides a logical order in which we introduce and discuss the existing work.

In this chapter we introduce the important concepts in NURBS curves and surfaces by reviewing the existing Spline Curve literature, in an approximately chronological order. Further, we extend some of the standard definitions to higher dimensional scenarios, with a view to applying it to the problem of modelling image manifolds. Next we apply our NURBS curves and surfaces to our example image manifolds, using these to generate unseen in-between images, the same experimental procedure we use throughout, and illustrated on page 143. Finally we conclude with a short discussion of these results and how they fit within the rest of this thesis.

9.2 Traditional NURBS background

9.2.1 Bézier Curves

Bézier curves are defined (Piegl and Tiller, 1997) usually as:

$$\underline{C}(u) = \sum_{i=0}^{n} B_{i,n}(u)\underline{P}_i \qquad 0 \leq u \leq 1, \tag{9.1}$$

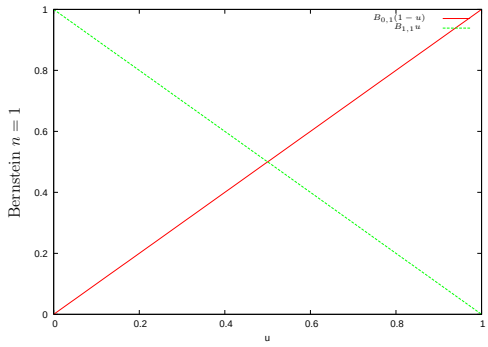
where $\underline{P}_i$ are the vector-valued control points which govern the shape of the curve and $B_{i,n}(u)$ are the basis functions, which are the n^{th} degree Bernstein polynomials given by:

$$B_{i,n}(u) = \frac{n!}{i!(n-i)!}u^i(1-u)^{n-i}. \tag{9.2}$$

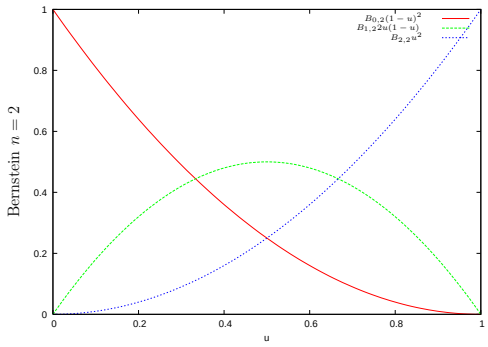
This definition provides the following properties which are advantageous to curve designers, and software developers. Firstly the control points $\underline{P}_i$ have an intuitive geometric relationship to the curve itself. Not only do the control points approximate the shape's curve, but further the convex hull of the control points will always contain the curve. Secondly curves are invariant under affine transformations. Finally the sum of the basis functions are normalised for u in the interval $[0, 1]$, that is $\sum_{i=0}^{n} B_{i,n}(u) = 1$. Furthermore the basis functions are symmetrical, and are numerically well-formed, which makes robust implementations a practical possibility. The Bernstein polynomials are illustated in Figure 9.1.

Figure 9.2 shows two examples of different degree Bézier curves. In this example the curves are defined by three and four control points $(n + 1)$ respectively. The curves start and end at the first and last control points, and the interior of the curves are influenced by the other control points.

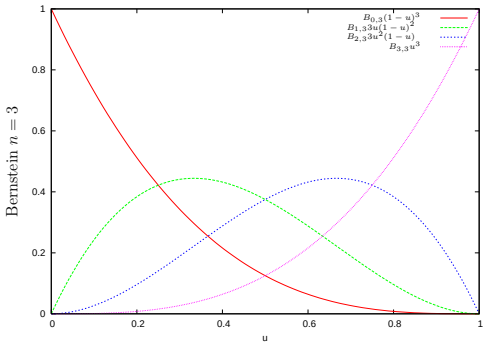
If we desire models of longer stretches of curve, with imposed conditions of continuity then it is possible to *blend* several Bézier curves to acheive this. In order to blend the curves smoothly (i.e. the tangents, as well as the positions, at the points $u_0 = 1$ and $u_1 = 0$ on the first and second curves respectively are



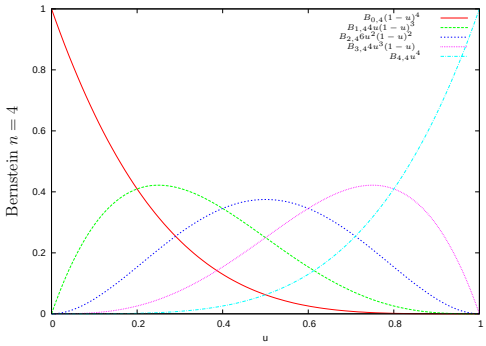
(a) 1st degree Bernstein polynomials



(b) 2nd degree Bernstein polynomials

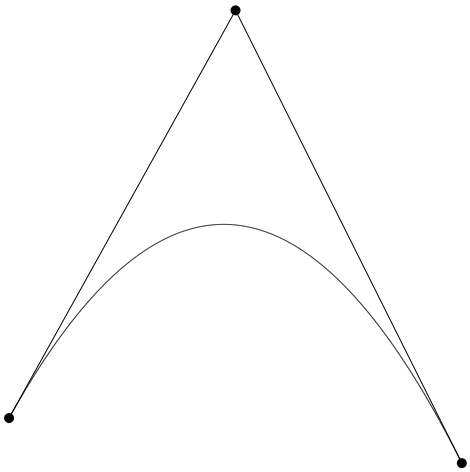


(c) 3rd degree Bernstein polynomials

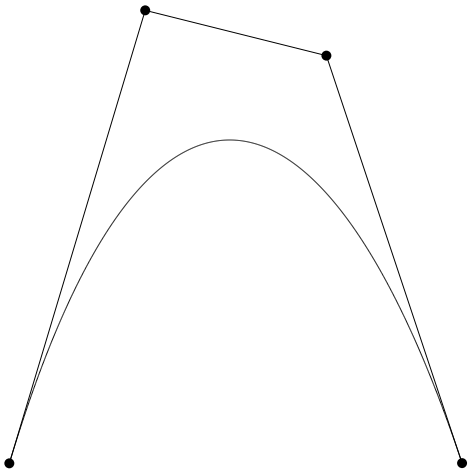


(d) 4th degree Bernstein polynomials

Figure 9.1: Bernstein polynomials



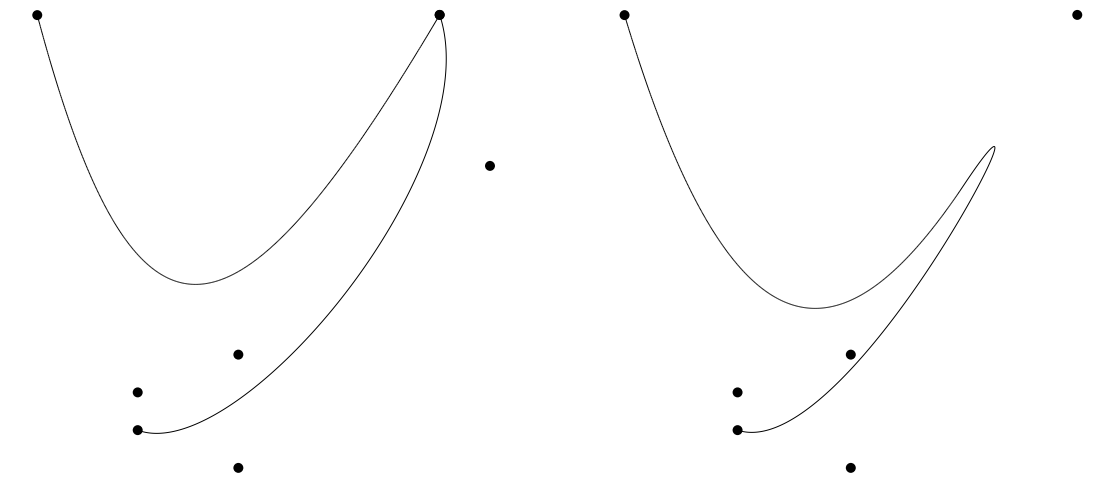
(a) Second degree Bézier curve



(b) Cubic Bézier curve

Figure 9.2: Bézier curve examples

equal) the vectors $\underline{P_{n-1}} - \underline{P_n}$ and $\underline{P_1} - \underline{P_n}$ must be equated (i.e. the direction at the start of one span must match the previous span). This is illustrated in Figure 9.3. For positional continuity, (Figure 9.3(a)) there was only one shared control point. For tangential continuity, in Figure 9.3(b) we now have two fewer control points than would be required for two 3$^{\text{rd}}$ degree curves, since two control points are now shared between the segments to meet the continuity requirements. Similarly even higher order continuity can be achieved with higher order curves by equating the vectors formed by the control points of the two curves further away from the point at which the two curves meet. In (Piegl and Tiller, 1997) this is referred to as a 'piecewise polynomial curve'.



(a) An example of the blending of two Bézier curves with positional continuity only

(b) Blending of two Bézier curves, with both positional and tangential continuity

Figure 9.3: Bézier curve blending

It is possible to use the definition of Bézier curves as the basis for higher dimensional analogs such as Bézier surfaces or Bézier hyper-surfaces, however this is relatively trivial and as we shall see in the next section we are more concerned with a generalisation of Bézier curves than Bézier curves themselves. Therefore a discussion of Bézier surfaces has been ommited in favour of a generalised discussion on surfaces later in Section 9.2.3.

9.2.2 NURBS curves

The Bézier curves which we have seen so far in this chapter are not however without limitations of their own. Firstly the effect of changing a single control point can be observed across the entire stretch of the curve. This non-locality of effect is often undesirable for curve designers for whom locality of effect is likely to be more intuitive. Secondly the only way to increase the number of constraints on the shape of the curve is to increase the degree of the curve which is often undesirable because of both the corresponding increase in processing required and the associated numerical instability it introduces. Thirdly the blended Bézier curves we saw previously are far from ideal for constructing longer stretches of curves — using blended curves requires the introduction of duplicated control points, which only serves to further confound the problems associated with the non-locality of the control points' influence.

Our motivations arise from these deficiencies. We desire a scheme where the influence of the control points is confined to only a relatively small segment of the curve and where the values of the control points do not influence the continuity of the curve at all, only its position. To this end we use variations on the scheme outlined here henceforth. We include an outline of a common form of this scheme, for further details the reader is referred to the comprehensive coverage to be found in (Piegl and Tiller, 1997), which was used as a basis for this description.

9.2.2.1 B-Spline curve definitions

$\underline{U}$ is defined as a *knot vector*:

$$\underline{U} = \{u_0, u_1, \ldots u_{m-1}, u_m\} \qquad u_i \leq u_{i+1} \forall i \in 0 \ldots m-1 \tag{9.3}$$

The u_i are referred to as *knots*. The basis functions ($N_{i,p}(u)$) can be defined in several ways, here we use the recursive definition, taken from (Piegl and Tiller,

1997), for the i^{th} basis function of a p^{th} degree B-spline:

$$N_{i,0}(u) = \begin{cases} 1 & \text{if } u_i < u < u_{i+1}, \\ 0 & \text{otherwise} \end{cases}, \tag{9.4}$$

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u). \tag{9.5}$$

Using the knot vector, and resulting basis functions we can define our curve to be the sum of the result of evaluating our basis functions for each control point:

$$\underline{C}(u) = \sum_{i=0}^{n} N_{i,p}(u) \underline{P}_i. \tag{9.6}$$

This formulation provides us with a flexible framework from which we can construct many different curves in many different ways. The result of this however is that when we come to build a curve we have to make a number of decisions about how to construct it.

Throughout this work we effectively use B-spline curves and surfaces, although we refer to our method as 'NURBS'. This is equivalent to fixing the w_i of (Piegl and Tiller, 1997, Equation 4.1) to be 1.

9.2.2.2 Knot vector selection

There are many different ways to construct B-Spline curves. In this thesis we use our curves and surfaces to model discretely sampled data. We therefore consider, in this section and Section 9.2.2.3 details of constructing a curve that are relevant to the scenario where a number of sampled data points $\underline{Q}_j$ are known. A curve is to be constructed, using these samples as an input to constrain it. In order to do this we first address the question of knot vector selection, and then in Section 9.2.2.3 consider the control points themselves.

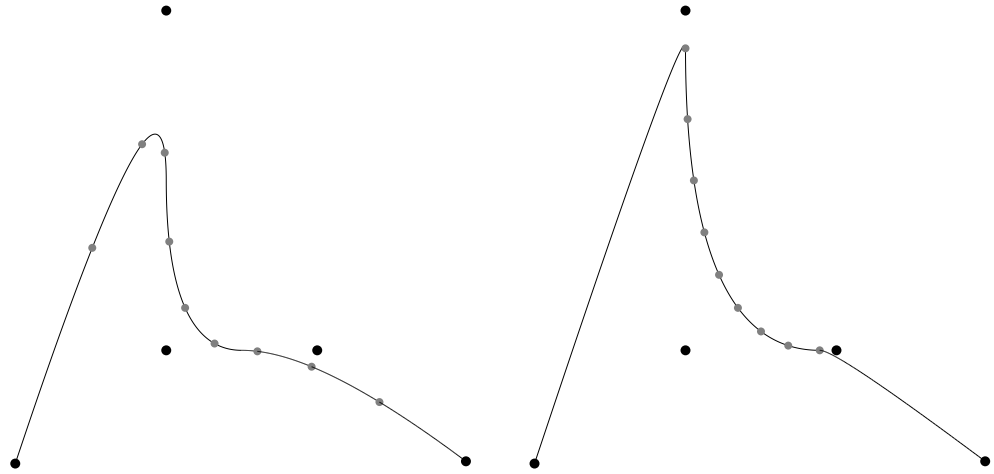
We previously mentioned that NURBS curves are a generalisation of Bézier curves. This can be seen by considering the knot vector $\{0, 0, 0, 0, 1, 1, 1, 1\}$ and

noting that the basis functions are now found to be the (3^{rd} degree) Bernstein polynomials. This holds true generally, Equation 9.7 shows the generalisation for a p degree Bernstein polynomial.

$$u_i = \begin{cases} 0 & \text{if } i < p+1, \\ 1 & \text{if } p+1 < i < 2(p+1) \end{cases} . \tag{9.7}$$

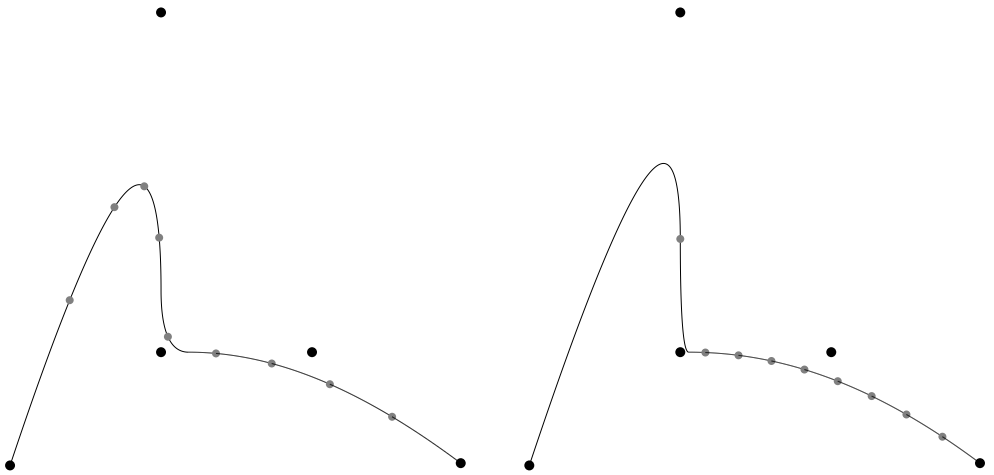
The effects of changing the knot vector on a NURBS curve can be dramatic, causing large changes to the shape of the curve. The knot vector itself can be thought to 'encode' the mapping between speed in the parameter space of the curve, and the corresponding speed in the space in which the curve is embedded. This is important when we model our image manifolds because we seek to construct models which represent both the speed and the position in image space. The effects of varying the knot vector in 2-D are illustrated in Figure 9.4. In this example the knot vector used is shown for each case, and there are nine light grey dots which are evenly spaced (0.1 between each one) in parameter space. Here we can see, for example, that in Figure 9.4(d), where the two interior knots are close to 1 that most of the parameter space of the curve is represented by the final segment of the curve. Conversely in Figure 9.4(b) most of the parameter space is represented by the middle of the curve. This also has the effect of 'pulling' the curve closer to the top (2^{nd}) control point, because the 3^{rd} control point does not have as much influence over the curve near the start.

Once one has understood the basic effects of changes to the knot vector the important issue to consider is how to effectively choose a knot vector when trying to construct a surface based on sample points. The most basic strategy would be to equally space each knot, resulting in an evenly spaced traversal of the curve it produces. Equation 9.8 shows an evenly spaced knot, for a p^{th} degree curve with



(a) $\underline{K} = \{0, 0, 0, \frac{1}{3}, \frac{2}{3}, 1, 1, 1\}$ (uniform)

(b) $\underline{K} = \{0, 0, 0, \frac{1}{10}, \frac{9}{10}, 1, 1, 1\}$



(c) $\underline{K} = \{0, 0, 0, \frac{9}{20}, \frac{11}{20}, 1, 1, 1\}$

(d) $\underline{K} = \{0, 0, 0, \frac{17}{20}, \frac{9}{10}, 1, 1, 1\}$

Figure 9.4: The same control points on a 2^{nd} degree NURBS curve with a selection of knot vectors

m sample points.

$$u_0 = \cdots = u_p = 0, \quad u_{m-p} = \cdots = u_m = 1,$$
$$u_{j+p} = \frac{j}{n-p+1}, \qquad j = 1, \ldots, n-p. \tag{9.8}$$

According to Piegl and Tiller (1997) this choice often proves to be a poor one and can lead to further problems during the computation of suitable control points. Instead they recommend the following method, which they term *averaging*, where $\bar{u}_i$ are the parameter values of the corresponding sample $\underline{Q}_i$:

$$u_0 = \cdots = u_p = 0 \quad u_{m-p} = \cdots = u_m = 1 \tag{9.9}$$
$$u_{j+p} = \frac{1}{p} \sum_{i=j}^{j+p-1} \bar{u}_i \qquad j = 1, \ldots, n-p \tag{9.10}$$

In the context of our image manifold models however we already have a natural parametrisation available to us. We can opt to use simply what we know about the parametrisation of the camera or lighting which we used to acquire the samples, to produce a suitable knot vector. In the ideal case, where our sampling worked flawlessly, and was meant to be uniform this would be a uniform knot vector. In practice we are usually suitably close to this ideal, and given that both the Linear model of Chapter 8 and the PDE model of Chapter 10 both (out of technical necessity) assume this to be the case it seems only fair that, for now at least, we also apply this same view here.

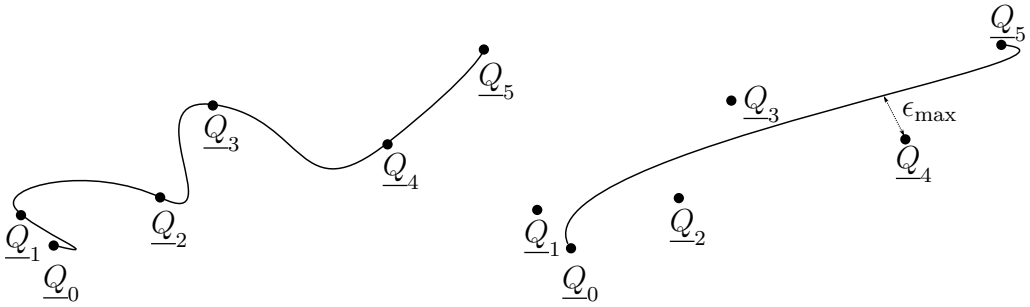
9.2.2.3 Control point selection

Control point selection is a key decision for any NURBS curve. Clearly the values chosen as the control points influence the shape of the curve heavily. It should further be noted that many common constructions of NURBS curves are such that the curve does not pass through all, if any, of the control points. As a result of

this we need to give some consideration to our choice of control points, and their relation to any sample data points $\underline{Q}_i$. Simply using the sample points $\underline{Q}_i$ as the control points $\underline{P}_i$ is generally not appropriate, for all but the lowest degree curves[1], since our sample points represent known positions of the curve and the curve will not pass through all of its control points.

The remainder of this section considers the case where several sample points are to be used in the construction of a NURBS curve, and outlines the possible approaches.

One of the key issues here when trying to use some sample points as the input to a control point selection scheme is deciding what the desired outcome should be. Should the resulting curve *interpolate*, that is pass exactly through each sample point, or should it instead *approximate*, whereby the curve passes near the sample points in some 'best-fit' scheme. This issue is illustrated in Figure 9.5, where (a) interpolates between the sample points, and (b) approximates to a given maximum distance, $\epsilon_{\max}$ between sample points and constructed curve.



(a) NURBS curve constructed to interpolate the samples $\underline{Q}_i$

(b) NURBS curve constructed to approximate the samples, $\underline{Q}_i$ to a given maximum tolerance $\epsilon_{\max}$

Figure 9.5: Interpolation of the sample points $\underline{Q}_i$ vs. Approximation

In the context of image manifolds this issue is of great importance. However, given that what we have seen in Chapter 8, and will be seeing in Chapter 10, are both interpolation we will defer discussion of approximation until later on, in Chapter 12. This makes sense from the point of view of both a straightforward

[1]This would be equivalent to the Linear model of the previous chapter anyway.

comparison of the other methods, and the discussion of the NURBS curves and surfaces themselves.

Here we consider a global approach to curve interpolation. Given an appropriate knot vector (see Section 9.2.2.2 for suggestions on this) we can formulate a suitable system of linear equations where $\underline{P}_i$ are our unknowns:

$$\underline{Q}_j = \sum_{i=0}^{n} N_{i,p}(\bar{u}_k)\underline{P}_i \tag{9.11}$$

By used Equation (9.9) to calculate a suitable knot vector we are now in a position to construct a system of linear equations using the $N_{i,p}(\bar{u}_k)$. The resulting linear system has several properties which make it well suited to solution by Gaussian elimination, notably it has a semi-bandwidth less than p. We use a standard third-party routine here to solve these equations.

In (Piegl and Tiller, 1997) the authors recommend avoiding Equation (9.8), to reduce the risk of the resulting system being singular. However in the case of our image manifolds, we already know the parametrisation of the curves and surfaces, which we assume to be uniformly spaced anyway. This causes averaging (the recommended method of avoiding Equation (9.8)) to be identical to evenly spaced knot vectors. To further confound the situation there is no single obvious interpretation of averaging in the n dimensional case.

9.2.3 NURBS surfaces

We have previously discussed NURBS curves, and assumed that they are embedded in a 2-D space, however there is really no need to constrain ourselves like this. We have provided examples through this context which are simpler to illustrate and explain, but the basic definition of a NURBS curve (Equation 9.6) can be extended trivially to cover a 2-D surface embedded in a 3-D space:

$$\underline{S}(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,q}(v)\underline{P}_{i,j}, \tag{9.12}$$

where n and m are the number of control points for the surface in the u and v directions, p and q are the degrees of the surface, and $N_{i,p}$ and $N_{k,q}$ are the B-Spline basis functions.

Notice how here we use the product of the basis functions (Equation 9.4), with independant knot vectors and degree for each dimension of the surface. This will have the effect of increasing the complexity of the control point and knot vector selection methods we discussed previously in Sections 9.2.2.3 and 9.2.2.2.

In order to choose a suitable set of control points $\underline{P}_{i,j}$, we construct a global surface that interpolates our sampled points. In (Piegl and Tiller, 1997) the authors note that by 'sweeping' the surface we can avoid constructing the full (n^2) system of equations we used to interpolate curves. Using the algorithm of (Piegl and Tiller, 1997, A9.4) we can pick our control points such that the created surface passes through each of our input points, exactly as we did for the curve in 2-D space. In the context of image manifolds this will not be discussed further here however, as we shall see in the next section.

9.3 Beyond surfaces in 3-D space

Until this point all of the examples we have considered, and literature we have cited, only considers low dimensional curves and surfaces, i.e. 2-D surface in 3-D space or 1-D line in 2-D or 3-D space. In the context of image manifolds this is not particularly useful. We need not be limited like this however, and indeed to extend NURBS further is not of itself novel, as several authors in the literature have touched upon this idea previously, although references to people using more than 6-D (extrinsic) NURBS surfaces are almost non-existent. The most notable NURBS work in greater than 3 dimensions is that of (Boeing Information and

Support Services, n.d.)[2], which has a number of citations in various forms. Other instances of n-dimensional NURBS found in the literature include (Cheatham et al., 2005), where the authors use NURBS to directly control 5 and 6 axis CAM systems. NURBS has also frequently been used in the literature to represent volumetric data, from MRI scans to atmospherics and optics (Martin and Cohen, 2001; Tustison and Amini, 2004; Wang and Jiang, 2007). The successes reported in these papers bode well for our attempts to extend and generalise the NURBS surfaces we have seen to this point for our eventual aim of modelling image manifolds.

In order to permit modelling of arbitrary, n-dimensional manifolds we are more concerned with generalised higher dimensional analogs of what we have previously covered. We begin this by extending our previous definition to add an extra parameter. Now, in the case of manifolds controlled by three parameters we extend our B-Spline surface definition to the following:

$$\underline{S}(u,v,w) = \sum_{i=0}^{n}\sum_{j=0}^{m}\sum_{k=0}^{o} N_{i,p}(u)N_{j,q}(v)N_{k,r}(w)\underline{P}_{i,j,k}. \tag{9.13}$$

This can be generalised further still:

$$\begin{aligned}\underline{S}(u_0,u_1,\ldots u_{n-1},u_n) \quad = \quad & \sum_{i_0=0}^{n_0}\sum_{i_1=0}^{n_1}\cdots\sum_{i_{n-1}=0}^{n_{n-1}}\sum_{i_n=0}^{n_n} N_{i_0,p_0}(u_0)N_{i_1,p_1}(u_1)\ldots \\ & \ldots N_{i_{n-1},p_{n-1}}(u_{n-1})N_{i_n,p_n}(u_n)\underline{P}_{i_0,i_1,\ldots,i_{n-1},i_n}. \end{aligned} \tag{9.14}$$

To compute the control points in this case we could use an adapted version of (Piegl and Tiller, 1997, A9.4), which is recursively called, in the same way A9.4 uses the curve interpolation algorithm many times. From a purely pratical standpoint however it proves to be easier to implement, and perfectly acceptable to use the naïve approach — forming a system of linear equations which we can solve to find control points which make the manifold pass through all the known sample points. We do this recursively to enable construction of any n-D object

[2]Unfortunately the website hosting this has been offline for the entire duration of my PhD.

embedded in an m-D space.

9.4 Experimental setup

There are many variations on NURBS that we have not discussed or considered here. For the purposes of this work a relatively simple, traditional NURBS implementation (albeit n-Dimensional, with a recursive implementation) represents the best approach for the remainder of this chapter. Were one to be building a production application as opposed to a proof of concept prototype then many of the other works we have not discused (e.g. DTNURBS (Boeing Information and Support Services, n.d.)) may become more relevant. This chapter however is focused on applying the ideas and theory on image manifolds that we have discussed in the preceding chapters. To this end our implementation is simplistic and in most respects traditional. Our choices at this stage reflect the linear (Chapter 8) and PDE model (Chapter 10) we use, in order to ease comparison of results. This choice to keep the implementation simple has the further additional benefit of simplifying the validation of our implementation.

Inspite of our simple implementation we still have a number of choices to make. As with the experiments we ran for the linear model we will consider a number of different gaps between our input sampled points. In addition to this we will construct models with different degree curves, (2, 4, 6, ...). We will simplify things slightly by using the same degree for each dimension.

We also have the knot vector selection strategy to consider. As we mentioned previously with uniformly spaced data (which we are forced to assume in Chapter 10) averaging becomes the same as a uniform knot vector. At this stage we have not discovered any more suitable knot vector selection strategies either, so this will remain fixed for all these experiments. In Chapter 12 we introduce and discuss an alternative knot vector generation scheme.

The only other option to consider right now is the selection of the control

points themselves. We previously mentioned that using the images themselves as control points does not make much sense, but we include this as a strategy for our experiments to confirm this statement.

We will also be returning to the NURBS model again in Chapter 12, with a number of alternative strategies.

A table detailing all of the configurations considered is included in Appendix F.

9.5 Results

Since our experimental setup effectively amounts to an unguided search of the parameter space of the NURBS surface creation what we present here is an abridged version of our results, which highlights the 'best' results we identified from our search.

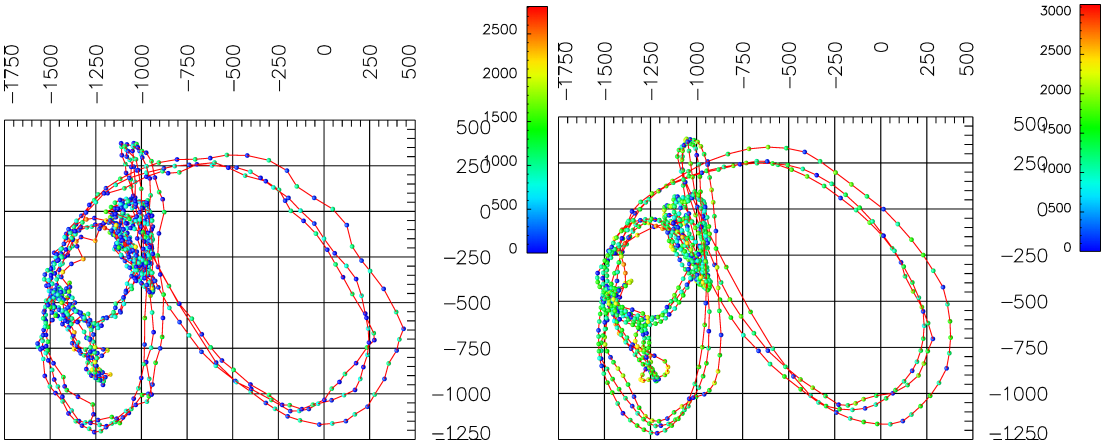
The definition of 'best' here is dependent upon our metric selection, which as we already noted, in Chapter 5, is application specific. The full results are shown in Appendix F.2 and are directly comparable to the results from the Linear model, as mentioned previously in Chapter 8 as well as the results from the PDE model in Chapter 10.

Once again our summary tables show mean error measure for all of the output images from our models.

9.6 Discussion

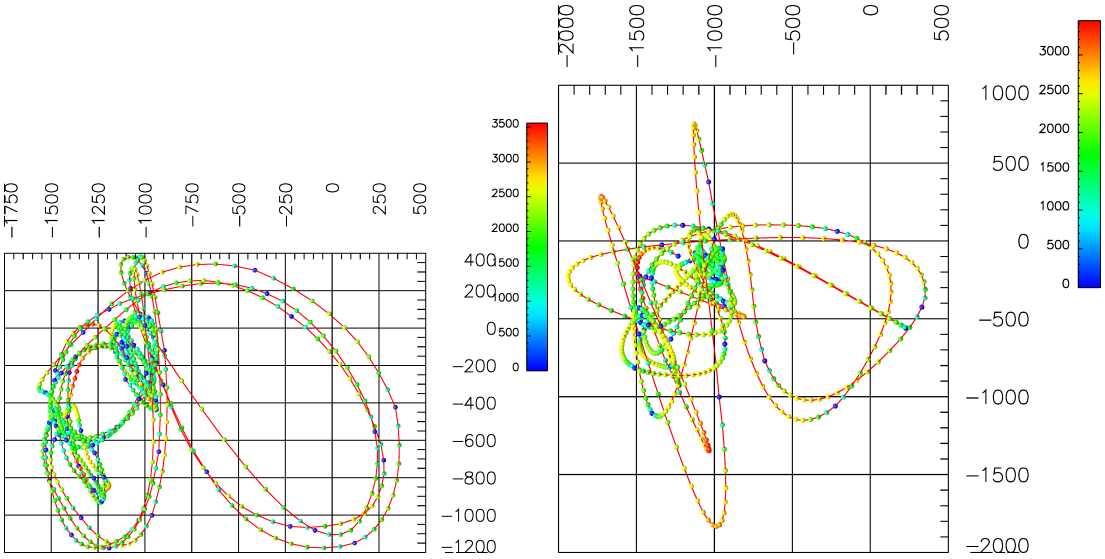
In this section we now look at a number of issues that these experiments raised. Primarily this discussion is focused exclusively on the NURBS model itself. We will not yet be looking at comparing the results with other models, until after we have introduced the PDE model as well.

In almost every single entry of the tables configuration number 1 (Table F.1, 391) won out. This is somewhat disappointing in most respects given that config-



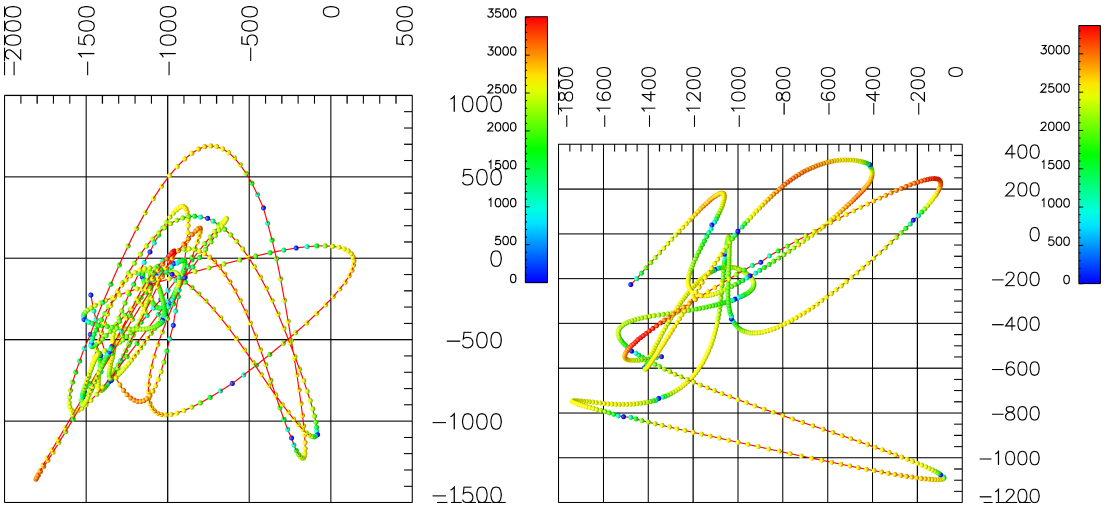
(a) $g = 2$

(b) $g = 5$



(c) $g = 10$

(d) $g = 20$



(e) $g = 25$

(f) $g = 50$

Figure 9.6: Visualisation of the detailed results for the CIRCLE_5 manifold, configuration 1

uration 1 uses a 2^{nd} degree curve, which was the lowest we tested. A 1^{st} degree curve would have been identical to the linear model. The only cases where this does not appear to be true is with the mutual information metric, with very large gaps (i.e. $g = 100$, Table F.4, page 399), where there is so little of the structure of the manifold represented as to make them useless anyway. This happens for instance with the CIRCLE_N dataset series.

Figure 9.6 shows configuration 1 applied to the CIRCLE_5 dataset as the gap increases. This corresponds to Figure 8.4 from the linear model. It is clear from this that our model does successfully construct smooth curves in image space. As with the linear model the structure appears to be rapidly lost at around $g = 20$. Between $g = 2$ and $g = 10$ the model seems to change less than seen in the linear model, i.e. the smoothness constraints appear to produce quite similar curves until the structure is damaged significantly.

This observation that the lowest degree curves produce the best results warrants further investigation. To further investigate this we have plotted degree against measured error, for a small selection of the datasets, using the Euclidean distance metric as an example. This plot is shown in Figure 9.7.

For the most part this plot seems to show that as we increase the degree of the curve (or surface) used to model the image manifold the error increases sharply until it levels off at some fixed value. The SINECOS manifold seems to be slightly different in this respect, but given that it does not reflect real images the unexpectedly high error for $p = 20$ is somewhat less interesting.

Figure 9.8 shows the PCA plots of the generated images for $p = 2$ with the 2DWOODBOX and STRAIGHT_5 manifolds. The results here look visually sensible, the 'smoothness' of the generated model is visible, if a little hard to spot in some areas because of the large gaps between neighbouring points. As would be expected the large gap in STRAIGHT_5, which was caused by lag during the capture process, causes by far the highest observed error of that dataset. In the case of

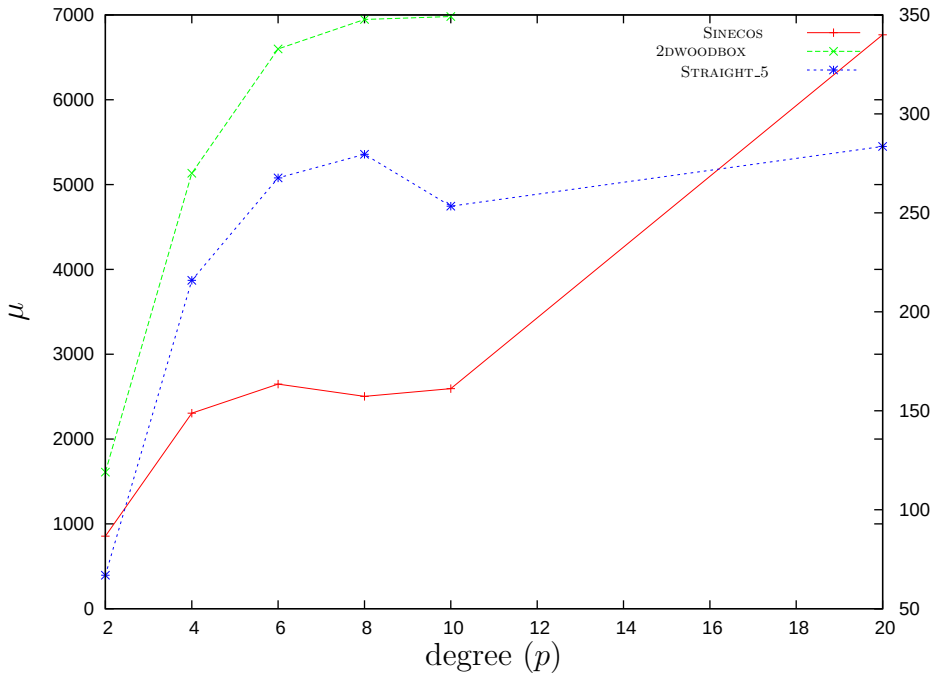
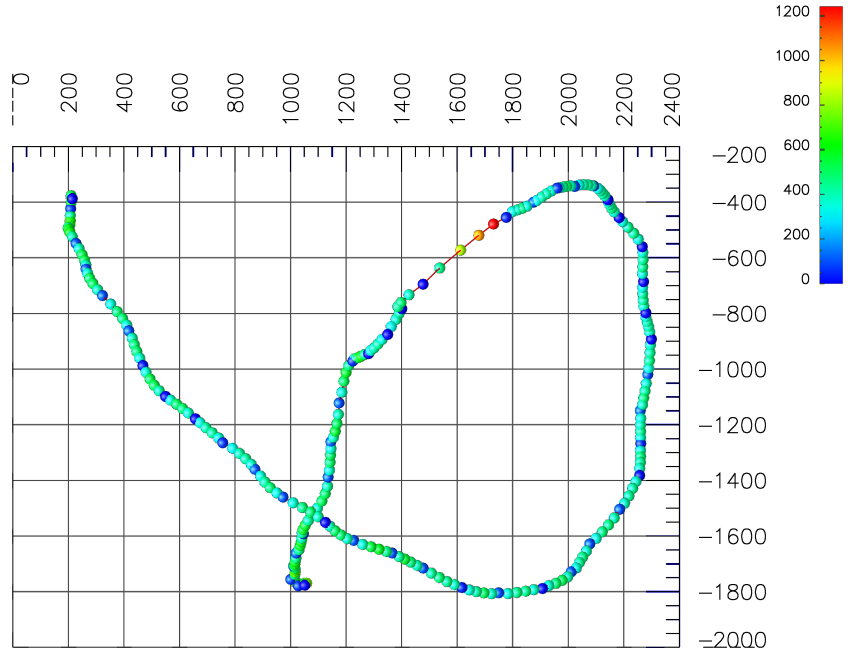


Figure 9.7: The effect of varying the degree of the curve/surface used, $g = 5$, SINCECOS on right axis, others on left

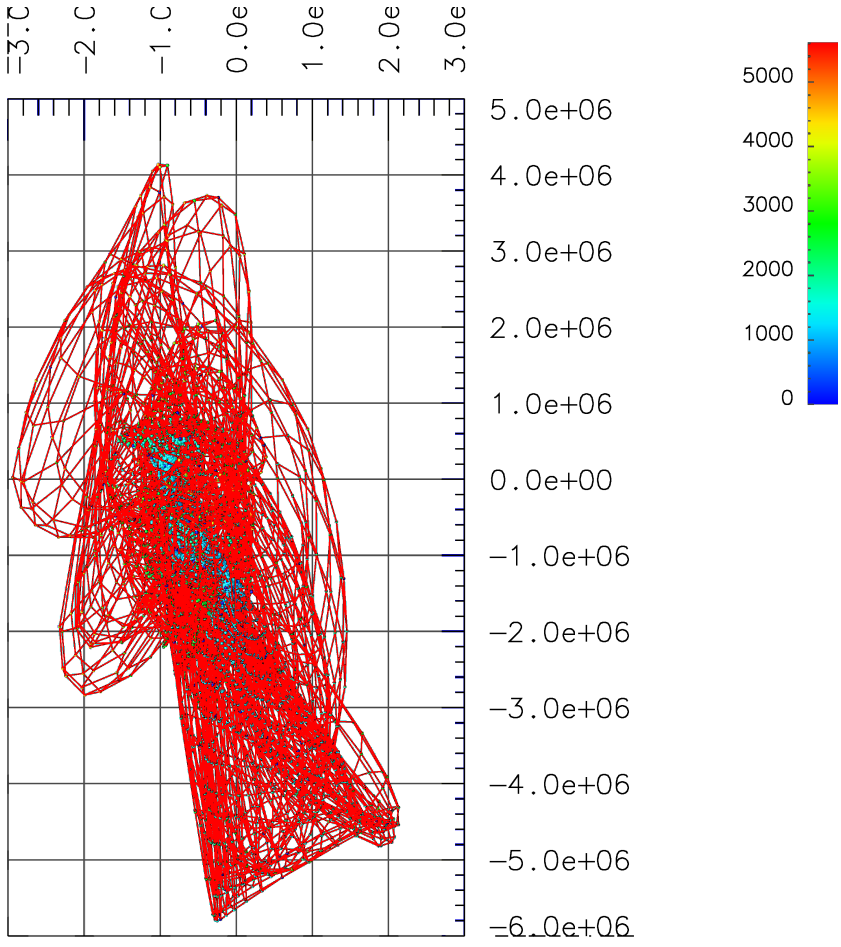
the 2DWOODBOX manifold it is hard to decipher what is actually happening in the centre of the bundle, and for that reason we now revert to parameter space plots, the corresponding one being shown in Figure 9.9.

In Figure 9.9 we can easily identify the two areas with the highest error. Both of these areas correspond to the passing of the opened ends of the box, close to the cameras. This makes sense geometrically given that this represents the point at which the view is changing most rapidly, because of the occlusions, and agrees with what we noted from the midpoint distance in Chapter 6.

The results for $p = 4$, which are shown in Figure 9.10, show some large errors around the edges of the surface. This goes some way to explaining why the 2$^{\text{nd}}$ degree curves out performed all the other configurations we considered. Figure 9.11 shows a sample image from each manifold illustrating the kind of errors in the actual images themselves. There is close to no discernible visual resemblance between these and the corresponding real images. What is visible in the images is not so much the errors that were introduced, but artifacts visible from clamping.



(a) Straight_5



(b) 2dwoodbox

Figure 9.8: PCA plots of the error (Euclidean) in generated manifold models for $p = 2$, $g = 5$

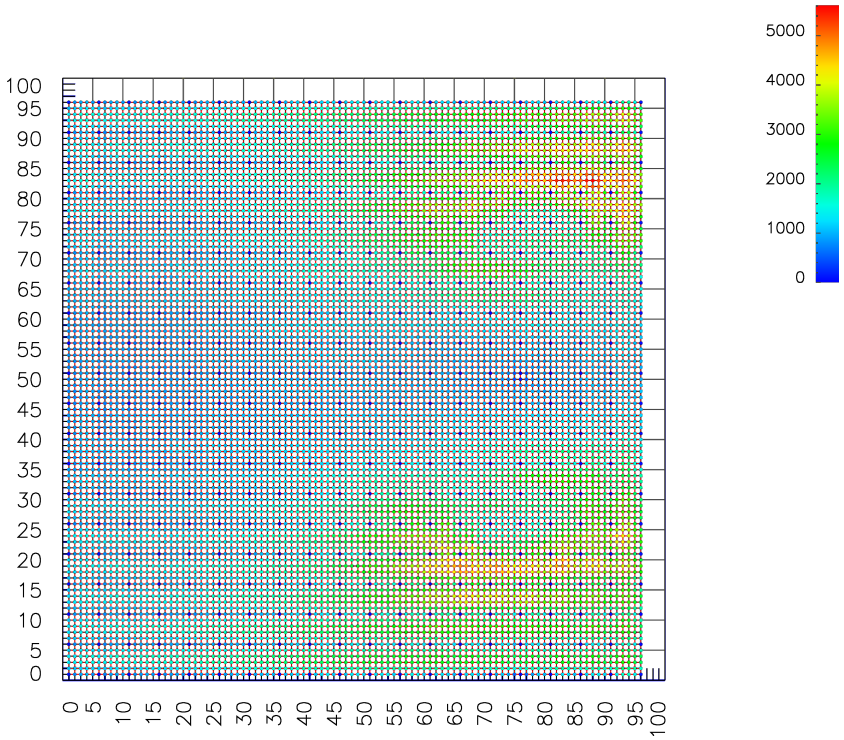
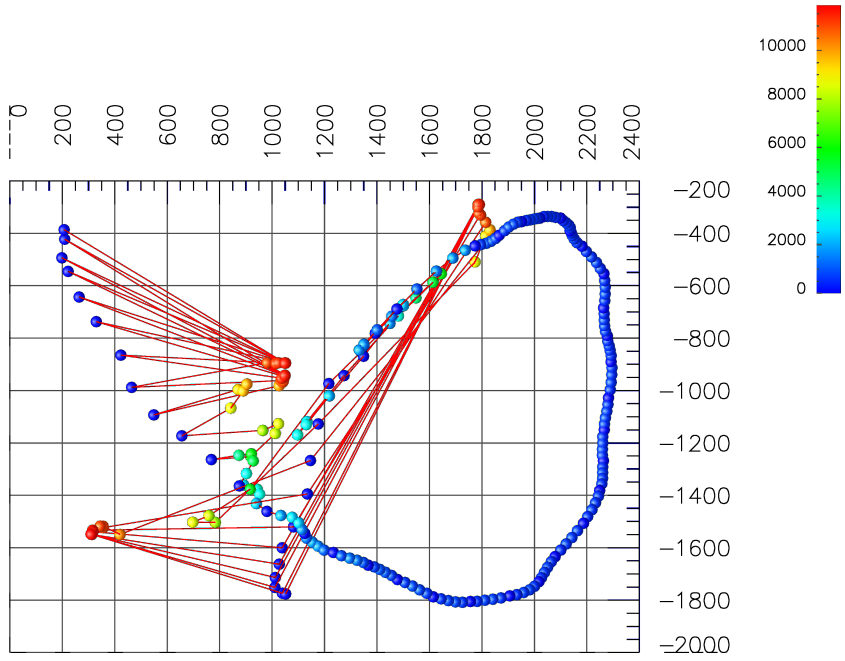


Figure 9.9: Parameter space plots of the error in generated manifold model of 2DWOODBOX for $p = 2$, $g = 5$

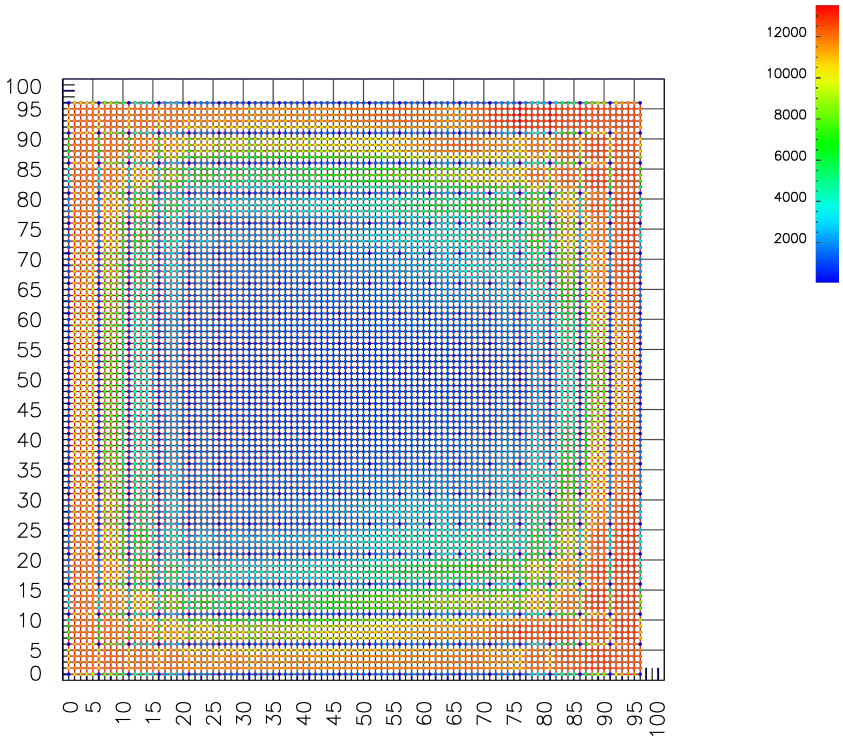
This will also have effected the positional and metric data. This clamping is being applied because many of the values returned by the model were greater than 10^5, which for something we expected to be in the $[0 : 255]$ range is very wrong!

In the centre of the manifolds (i.e. away from the edges for the 2DWOODBOX dataset) the image become more sensible, as though nothing was wrong. This trend is also observed more generally with the other datasets. As we increase the degree even further (not shown here), these results become worse, with the affected areas at the edges encroaching further into the centre, resulting in more, higher error images.

Finding the cause of this is an interesting problem. Essentially we now wish to establish whether or not this represents a problem with the whole concept of higher order models of image manifolds, or if it represents a problem with our implementation, our choice of knot vector, or alternatively if the lack of information about the derivatives at the end of the surface is problematic.

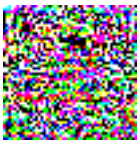


(a) (PCA) Straight_5



(b) (Parameter) 2dwoodbox

Figure 9.10: Plots of the error (Euclidean) in generated manifold models for $p = 4$, $g = 5$



(a) STRAIGHT_5

(b) 2DWOODBOX

Figure 9.11: Sample images from the $p = 4$ models

The first port of call in this investigation takes us back to the system of equations we generated to perform the global interpolation. Investigation shows that the 3rd party library we used to solve the system of equations failed to report any problems, and indeed found a solution that does pass almost exactly through each of the sample points it is supposed to interpolate. Given this it seems that the resulting models must then curve smoothly between points they interpolate, since the smoothness arises from the evaluation of the basis functions, and not the control points selected. We confirmed this statement experimentally by re-running the experiments and noting the outputs without the clamping. As well as producing the output images and measuring the distance, the experiments did also log various details about their run. This information did include both the knot vector (which seems perfectly sensible) and the system of equations used for the interpolation. Inspecting these equations reveals that, for the higher degree curves and surfaces at least, some of the coefficients in the matrix become very small, e.g. 5.1×10^{-6}, which may have contributed to what we have observed.

Since the small coefficients only occur when the model has large 'interior' areas, i.e. many more samples than the degree of the curve we can try a number of things. Firstly we propose and test a 'lightweight' local interpolation scheme, which offers no better continuity than G_1, simply building a larger number of smaller curves or surfaces. This allows us to neatly sidestep the possible problem with small coefficients in the matrix.

We tested this local approach on a small stretch of the STRAIGHT_5 manifold to see if it offered any prospect of improvement with the higher degree curve. Our

results are plotted in Figure 9.12, and a sample from the 4^{th} degree model is shown in Figure 9.13. Compared to what we observed in Figure 9.7 the increase in error is far less dramatic, although this is only a very limited number of data points. (Note that for the $p = 1$ case this is identical to the linear model of Chapter 8).

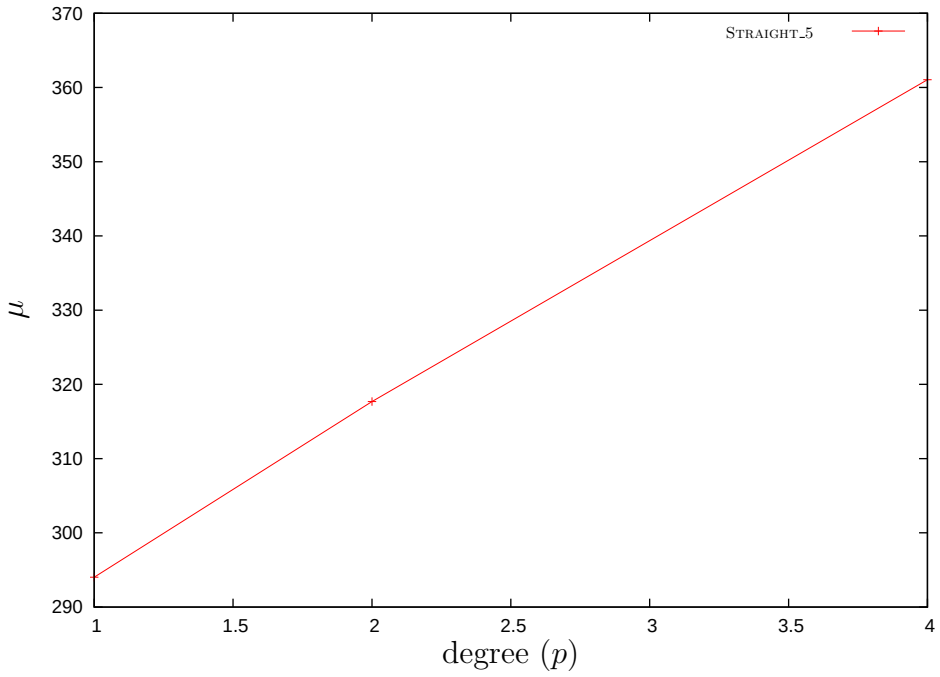
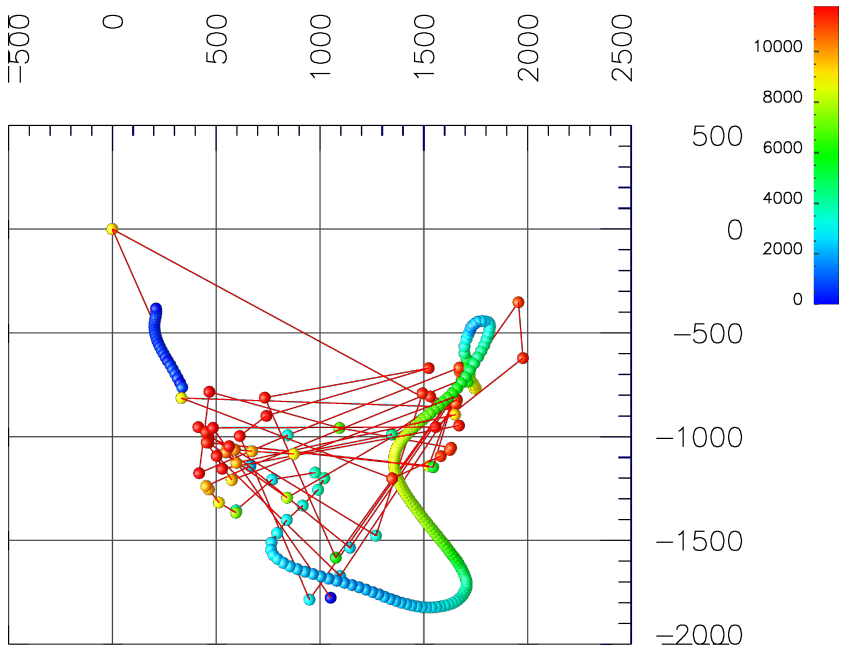


Figure 9.12: The effect of varying the degree of the curve, local interpolation test.

Figure 9.13: An example taken from the 4^{th} degree, local interpolation of a short stretch of STRAIGHT_5

This still leaves the question of why our higher order models failed when the interior of the surface is large. In (Piegl and Tiller, 1997) the authors suggested that the use of an equally spaced knot vector might lead to a singular system of equations. Of course, since our $\bar{u}$ is dictated and evenly spaced their recommended alternative method of averaging for the knot vector is identical anyway. Nonetheless it would still be interesting for us to try the same experiment, but with an alternative knot vector choice. Piegl and Tiller (1997) makes no further



(a) Straight_5

Figure 9.14: Alternative knot vector for the $p = 4$ model, Straight_5

suggestions for any other ways of choosing knot vectors, so we now produce a new knot vector, scaling it so as to shift more of the interior of the knot towards the 0 and 1 ends and have less coverage around the 0.5 area.

The results of this changed knot vector are quite predictable really — we move the area with the bad errors away from the start and finish of the manifold into the centre. The PCA plot of this is shown in Figure 9.14. We will return to the question of knot vector selection in Chapter 12.

This would suggest that we simply do not have enough constraints to successfully model higher order curves in this way. Since we are already using all the non-discarded data points one possible way of increasing the total number of constraints on the surface and forcing it to take a more sensible path in image space might be the use of derivative information along the surface itself. This information is unfortunately not available to us directly, and any estimates we produce at least from the current data would be based upon the assumption that the local curvature is well approximated by the global curvature, which is unlikely

to be true of the general case (Bichsel and Pentland, 1994; Donoho and Grimes, 2005). The results of the curvature measure we used in Chapter 6 indicated that locally the tangent estimates we produced were somewhat noisy. An investigation into the derivatives of image manifolds might however prove to be an interesting endeavour in the future, and we believe would potentially offer improvements to NURBS surface modelling. We suspect that the use of derivative information when building the curves and surfaces will help to construct better models, simply because they offer more, stricter constraints which is what seems to be lacking. We will also return to the subject of derivative estimates within the next chapter, in the context of PDE surfaces. It is also worth noting at this point that we will be returning briefly to the subject of NURBS in Chapter 12, where we address a number of possible ways of improving upon the results presented here, which do not rely upon derivative information.

9.7 Summary

In this chapter we have seen:

- a brief history and introduction to NURBS;
- how NURBS can work in higher dimensional spaces;
- how NURBS can work with general case n-manifolds, and not just curves and surfaces;
- an application of this to the concept of image manifolds;
- results of this applied to our selected image manifolds;
- a brief discussion of these results, and their implications for our work;
- a number of reasons why using derivatives might improve these results.

In the next chapter we will explore:

- the PDE surface method as an alternative to NURBS;
- the application of the PDE surface method to modelling image manifolds;
- results from the above.

Chapter 10

PDE method

10.1 Introduction

Historically, in computer graphics and surface modelling in general, NURBS has been seen as the normal approach and for this reason we have treated NURBS first in this thesis, in Chapter 9. Another, increasingly prominent alternative to NURBS is the notion of representing surfaces using Partial Differential Equations, or *PDEs*.

In part this increased prominence is due to the increase in computational power available to users, which has facilitated real-time implementations of surface viewers and other tools important to users. This is not the only reason by any means, much work has been devoted to developing improved techniques and algorithms which make PDE surfaces more intuitive and useful to work with, and permit some of their inherent properties to be better exploited.

The method of representing surfaces using partial differential equations, referred to as the *PDE surface method* henceforth, has been proposed (Bloor and Wilson, 1990; Ugail et al., 1999) as an alternative to methods such as NURBS (Chapter 9) in a typical CAD application. In this chapter we look at extending this method to model surfaces (or more generally manifolds) which are embedded in much higher dimensional spaces than required for a CAD application, making

it suitable for modelling image manifolds. Additionally we look at increasing the number of parameters used to specify a point on the manifold, which can be more than the typical (u, v) parametrisation of a surface used in a CAD application. One such existing example of using the PDE surface method to model a volume is presented in (Du and Qin, 2007).

In this chapter we will firstly review some of the existing PDE surface literature as well as some of the fundamental concepts involved. We will consider several different approaches to solving the resulting equations, and discuss these with a view to applying PDE surfaces to the problem of modelling image manifolds as we did with NURBS in the previous chapter.

10.2 Traditional PDE surfaces

There have been many different surface modelling techniques proposed in the past, in particular parametric surface modelling.

A parametric surface $\underline{X}(u, v)$ in 3D space can be expressed as

$$\underline{X}(u, v) = (x(u, v), y(u, v), z(u, v)) \qquad x, y, z \in \mathbb{R}. \tag{10.1}$$

By assuming our surface to be periodic in v, and restrict it to the finite domain $\Omega = \{u, v : 0 \leq u \leq 1; 0 \leq v \leq 2\pi\}$ we make a simple, efficient solution possible. It has been proposed (Ugail et al., 1999) we can represent this surface as an Elliptic PDE, we therefore have that:

$$\left(\frac{\partial^2}{\partial u^2} + \alpha^2 \frac{\partial^2}{\partial v^2}\right)^2 \underline{X}(u, v) = 0, \tag{10.2}$$

where α is a "smoothing parameter", which controls the length over which the boundary conditions influence the interior of the surface. This equation is often referred to as the *biharmonic*. This choice is sensible for two reasons: firstly it

permits an efficient and stable solution, suitable for online computations which we will outline in this chapter; secondly it is well suited to a number of common problems in the typical CAD application, for which this method was originally developed. Furthermore it fits well with a number of the datasets we are using in this thesis. Even in the cases where this does not naturally fit our datasets we are able to work around it later on.

In the remainder of this section we review one of the more commonly used existing solutions to this equation. Existing work using the PDE surface method (Ugail et al., 1999) has used the method of separation of variables (Churchill and Brown, 1978) to find an analytic solution to the PDE above, for which a computationally efficient implementation is also possible. We also assume that sufficient boundary conditions have been imposed upon the surface. The basic solution to this is now outlined.

Given suitable boundary conditions

$$\begin{aligned} \underline{X}(0,v) &= f_0(v), \\ \underline{X}(1,v) &= f_1(v), \\ \underline{X}_u(0,v) &= g_0(v), \\ \underline{X}_u(1,v) &= g_1(v), \end{aligned} \tag{10.3}$$

which we assume are continuous and closed, and given that Equation 10.2 is elliptic we can use separation of variables (Churchill and Brown, 1978) to obtain the following general solution:

$$\underline{X}(u,v) = \underline{A}_0(u) + \sum_{n=1}^{\infty} \left[\underline{A}_n(u)\cos(nv) + \underline{B}_n(u)\sin(nv)\right], \tag{10.4}$$

where $\underline{A}_0$, $\underline{A}_n$ and $\underline{B}_n$ are vector valued functions as follows:

$$\begin{aligned}
\underline{A}_0 &= \underline{a}_{00} + \underline{a}_{01}u + \underline{a}_{02}u^2 + \underline{a}_{03}u^3, \\
\underline{A}_n &= \underline{a}_{n1}e^{\alpha nu} + \underline{a}_{n2}ue^{\alpha nu} + \underline{a}_{n3}e^{-\alpha nu} + \underline{a}_{n4}ue^{-\alpha nu}, \\
\underline{B}_n &= \underline{b}_{n1}e^{\alpha nu} + \underline{b}_{n2}ue^{\alpha nu} + \underline{b}_{n3}e^{-\alpha nu} + \underline{b}_{n4}ue^{-\alpha nu}. \qquad (10.5)
\end{aligned}$$

Fourier analysis allows us to write the boundary conditions in the following form, from which values of the constants in the general solution can be calculated. Fourier analysis of the boundary conditions gives us:

$$f_0(v) = \underline{c}_{00}(u) + \sum_{n=1}^{\infty} \left[\underline{c}_{n0}(u)\cos(nv) + \underline{d}_{n0}(u)\sin(nv)\right], \qquad (10.6)$$

$$f_1(v) = \underline{c}_{01}(u) + \sum_{n=1}^{\infty} \left[\underline{c}_{n1}(u)\cos(nv) + \underline{d}_{n1}(u)\sin(nv)\right], \qquad (10.7)$$

$$g_0(v) = \underline{c}_{02}(u) + \sum_{n=1}^{\infty} \left[\underline{c}_{n2}(u)\cos(nv) + \underline{d}_{n2}(u)\sin(nv)\right], \qquad (10.8)$$

$$g_1(v) = \underline{c}_{03}(u) + \sum_{n=1}^{\infty} \left[\underline{c}_{n3}(u)\cos(nv) + \underline{d}_{n3}(u)\sin(nv)\right]. \qquad (10.9)$$

This allows us to find values of our constants $\underline{a}_{nm}$ and $\underline{b}_{nm}$ for each dimension and Fourier mode:

$$\begin{pmatrix} \underline{a}_{n0} \\ \vdots \\ \underline{a}_{n3} \end{pmatrix} = M \begin{pmatrix} \underline{c}_{n0} \\ \vdots \\ \underline{c}_{n3} \end{pmatrix}, \qquad (10.10)$$

$$\begin{pmatrix} \underline{b}_{n0} \\ \vdots \\ \underline{b}_{n3} \end{pmatrix} = M \begin{pmatrix} \underline{d}_{n0} \\ \vdots \\ \underline{d}_{n3} \end{pmatrix}, \qquad (10.11)$$

where M is the matrix:

$$\begin{bmatrix} 1 & 0 & 1 & 0 \\ e^{\alpha n} & e^{\alpha n} & e^{-\alpha n} & e^{-\alpha n} \\ \alpha n & 1 & -\alpha n & 1 \\ \alpha n e^{\alpha n} & e^{\alpha n}(1+\alpha n) & -\alpha n e^{-\alpha n} & e^{-\alpha n}(1-\alpha n) \end{bmatrix}, \qquad (10.12)$$

which was calculated by considering the boundary conditions at $u = 0$ and $u = 1$. This can be very fast to solve — we have a very small linear system to, which can be solved directly and a set of Fourier transforms, for which there are numerous highly optimised implementations of highly efficient algorithms available. This solution is however not the only solution we will be using; as we shall see later on in this chapter, in cases where we have more than two independent variables we will be forced to use an alternative solution.

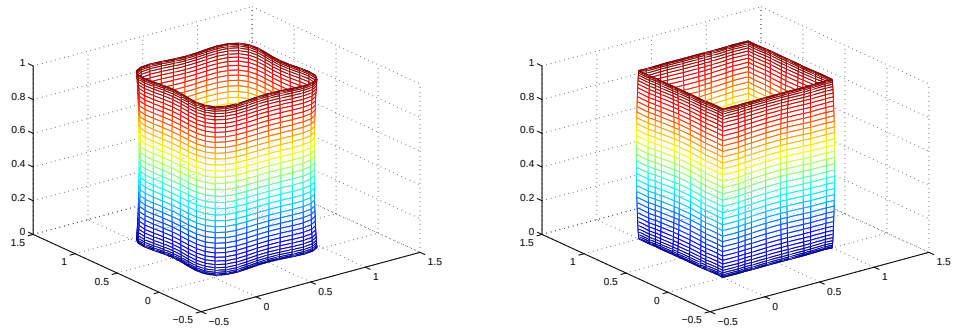
In practice not all boundary conditions have a Fourier series as simple as that of a circle or ellipse. This means that our boundary conditions cannot always be represented by a short, or even finite Fourier series. One such example is shown in Figure 10.1. For this reason the Fourier series used in Equation 10.4 is commonly truncated at some given value of n, typically $N = 6$ as in (Ugail et al., 1999). A remainder term can be introduced to approximate the higher order components of the boundary condition that would otherwise be lost:

$$\underline{X}(u,v) = \underline{A}_0(u) + \sum_{n=1}^{N} [\underline{A}_n(u)\cos(nv) + \underline{B}_n(u)\sin(nv)] + \underline{R}(u,v). \tag{10.13}$$

$\underline{R}(u,v)$ is usually chosen to be of the form

$$\underline{R}(u,v) = r_0(v)e^{\omega u} + r_1(v)e^{-\omega u} + r_2(v)ue^{\omega u} + r_3(v)ue^{-\omega u}, \tag{10.14}$$

where $r_0(v), \ldots, r_3(v)$ are calculated by considering the difference between the real boundary conditions, and the approximation of them after truncation of the Fourier series. Choosing $\omega = \alpha(N+1)$ gives a decay rate similar to the decay rate of the actual solution given that the mode $(N+1)$ is the dominant mode of the modes that have been truncated.



(a) The truncated Fourier series poorly represents the square boundary condition

(b) With a remainder term the boundary conditions are much squarer

Figure 10.1: An example of the remainder term, square boundary conditions with $N = 5$

10.3 Numerical Methods for PDE surfaces

In addition to the previously outlined solution it is also possible to solve such PDEs using numeric methods. In order to do this a suitable discrete scheme must be derived. The basic premise of this solution is that the domain Ω may be mapped onto a discrete representation of the domain. Within this discrete domain it is then possible to formulate a solution to the PDE whereby each point in the new discrete domain is a function of its neighbours. This in turn will result in a large linear system of equations that can be solved via an appropriate method, discussed later on in this chapter.

Presented in this section is the approach we take to using numerical methods to solve elliptic PDEs, as such it is kept as relevant as possible to the context they will be used in this thesis. For a more complete discussion of numerical solutions to PDEs the reader is referred to the text by (Thomas, 1995).

10.3.1 Laplacian

In order to address the issue of numerical solutions it makes sense to first consider a simpler PDE than the biharmonic of Equation 10.2. Equation 10.15 shows what is usually referred to as the *Laplacian*, a 2^{nd} order elliptic PDE instead of the 4^{th}

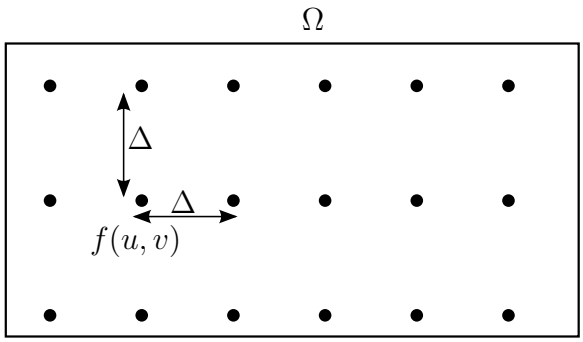


Figure 10.2: Discretisation of the domain Ω

order biharmonic previously discussed

$$\left(\frac{\partial^2}{\partial u^2} + \frac{\partial^2}{\partial v^2}\right) \underline{X}(u, v) = 0. \tag{10.15}$$

Firstly we discretise our domain Ω and produce a finite grid (Figure 10.2) that represents it. Then we seek a discrete approximation in terms of the grid of the various derivatives involved. We assume here that the grid has the same resolution in all directions to simplify the problem, although this need not always be the case. In this case we have used a centred difference scheme, Equation 10.16, because it is a more accurate approximation of the derivatives. See (Thomas, 1995) for further information on different possible schemes.

$$\frac{\partial}{\partial u} \approx \frac{1}{\Delta}\left[f(u + \frac{\Delta}{2}, v) - f(u - \frac{\Delta}{2}, v)\right] \tag{10.16}$$

Using this discrete approximation of $\frac{\partial}{\partial u}$ and a similar one corresponding to $\frac{\partial}{\partial v}$ we can produce two further discrete approximations:

$$\frac{\partial^2}{\partial u^2} \approx \frac{1}{\Delta^2}\left[f(u + \Delta, v) - 2f(u, v) + f(u - \Delta, v)\right], \tag{10.17}$$

and similarly:

$$\frac{\partial^2}{\partial v^2} \approx \frac{1}{\Delta^2}\left[f(u, v + \Delta) - 2f(u, v) + f(u, v - \Delta)\right]. \tag{10.18}$$

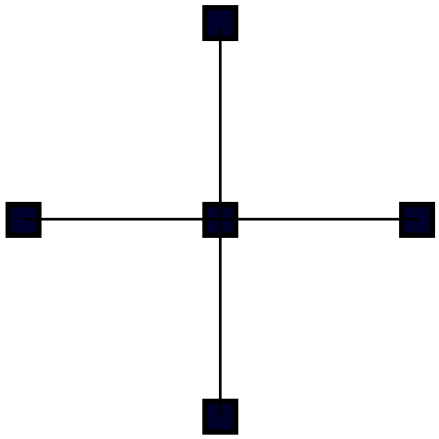


Figure 10.3: Visual illustrations of the 2nd order 2-D scheme

By substituting Equations 10.17 and 10.18 into Equation 10.15 we get a discrete scheme, illustrated in Figure 10.3, for the Laplacian:

$$\frac{1}{\Delta^2}\left[f(u+\Delta,v)+f(u-\Delta,v)+f(u,v+\Delta)+f(u,v-\Delta)-4f(u,v)\right]=0. \tag{10.19}$$

By applying this scheme to the entire discrete domain, filling in the specified boundary values at the boundaries, we get a large system of linear equations. The system has n^2 unknowns, x_i, comprised of the points in the discrete grid, and the right hand side, b_i, either 0 for interior points on the grid or some constant, given by the boundary conditions on or near the edges of the grid. The system of

equations is of the form:

$$\begin{bmatrix} \ddots & & & & & & & & \\ & \ddots & & & & & & & \\ & & \ddots & & & & & & \\ & & & \ddots & & & & & \\ & 1 & \dots & 1 & -4 & 1 & \dots & 1 & \\ & & & & & \ddots & & & \\ & & & & & & \ddots & & \\ & & & & & & & \ddots & \\ & & & & & & & & \ddots \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ x_{n^2-1} \\ x_{n^2} \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ b_{n^2-1} \\ b_{n^2} \end{bmatrix}, \tag{10.20}$$

Having produced a suitable linear system of equations that approximates the solution to our given PDE we now seek an efficient solution of that system. In all but the most trivial of cases any direct solution (e.g. inversion, or via some decomposition) is not feasible, simply because of the size of it. Therefore we turn to an iterative approach, which we discuss briefly here, but for further details the reader is referred to (Varga, 1962).

We start by applying the Jacobi iterative formula, which results in the following equation:

$$f^k(u,v) = \frac{1}{4}\left[f^{k-1}(u-\Delta,v) + f^{k-1}(u,v-\Delta) + f^{k-1}(u+\Delta,v) + f^{k-1}(u,v+\Delta)\right], \tag{10.21}$$

where $f^k(u,v)$ denotes the value of the point on the grid u,v after the k^{th} iteration.

By definition we know that Equation 10.19 should equal zero. This gives us a natural error function which we can then apply to control the iterative process,

iterating until the maximum residual error (ϵ) is below some given threshold[1]:

$$\epsilon(u,v) = |f^{k-1}(u,v-\Delta)+f^{k-1}(u,v+\Delta)+f^{k-1}(u-\Delta,v)+f^{k-1}(u+\Delta,v)-4f^{k}(u,v)|. \tag{10.22}$$

At the boundaries of our grid we use the boundary conditions, Equation 10.3, specified to supply fixed, known values. Initial values for the rest of the grid are also important, and poorly chosen initial values can easily cause the solution to take much longer to converge. A simple solution might be to pick the average of all the supplied boundary values, or to use simple linear interpolation in the interior.

Notice here that the Jacobi iterations requires the retention of the f^{k-1} values until all f^k values have been computed. This doubles the amount of memory required to hold the working set of a naïve implementation. Using Gauss-Seidel relaxations does away with this, allowing new values to be computed 'in-place'. This improves the efficiency of the algorithm somewhat, and the only minor complication in this is with calculating the residual error. To that end we use the new error function:

$$\epsilon(u,v) = \left| \frac{f^{k}(u,v) - f^{k-1}(u,v)}{f^{k}(u,v)} \right|. \tag{10.23}$$

This error function measure the magnitude of the error that was removed by the iteration. If the solution has exactly converged then $f^k(u,v) - f^{k-1}(u,v)$ will be 0, otherwise it will be some value indicating how close we are to convergence. The general convergence properties of this method are not discussed in depth here, but are well covered in the literature (Thomas, 1995). For our purposes the convergence or non-convergence of a particular case is obvious enough empirically, meaning that the run-times will be high (or infinite), unless constrained otherwise.

[1] We will discuss this threshold in relation to our manifold models later on in this chapter.

10.3.2 Multigrid solutions

Even the Gauss-Seidel relaxations are still not very efficient (Briggs et al., 2000) for large grids however, and so we now turn to multigrid solutions to improve things further. As with the rest of the theory in this chapter a brief overview of the areas relevant to this thesis is given here, but for a more detailed discussion the reader is referred to (Briggs et al., 2000).

The basic premise of a multi-grid solution is that the solution is computed at various different scales to provide a better initial estimate. In effect this is just the logical conclusion of what we said earlier, in Section 10.3.1, about the impact of poor choices for starting values upon the time taken to converge. Can we do better than the simple estimates we used? Absolutely, but the trap to avoid is spending so much time on computing better estimates that we end up dominating the time taken to compute the actual solution!

The answer to this is to calculate the solution on not just one single grid, but on a number of different resolution grids. A solution on a very small grid takes much less time to compute than on the full-size grid. Likewise a medium scale grid takes more time to compute than on a small grid, but less than a full grid would. It happens as well that the small scale grid makes a good approximation of the solution on the medium scale grid. By repeating this approach many times, and using the solution from the previous (i.e. nearest, smaller) grid as the starting point for the current grid we can be sure we have good starting values for all the positions on the grid, whilst remaining efficient in our solution.

If we look at this from another perspective we can see why this would work well. If all of the grid points within a given subset of the grid are very close to having no error (by either Equation 10.22 or 10.23) yet the whole error on the grid is significant then it implies these points are not close to the actual solution, they merely meet the local constraints. This however means that any cluster of points which has close to no error will change only very slowly, as the correct

values propagate across the whole grid from the boundary conditions. If however we had just a single point on the grid that was not correct then this would be quickly corrected by a relatively small number of iterations. This single error, and other small errors can be thought of as high frequency errors — if we performed spectral analysis of the grid these would show up as high frequency components. Conversely errors which affect the whole grid can be thought of as low frequency errors. With multigrid solvers we simply exploit the fact that a low frequency error on a large grid is a high frequency error on a smaller grid.

In order to implement multigrid we still need some iterative solver, like we used earlier. This is often termed a *relaxation operator*, or simply *smoothing operator*. Gauss-Seidel makes a sensible choice here, given the storage requirements. Additionally we need some method of transferring solutions between two grids of different resolutions. These are usually termed *restriction* and *prolongation* operators, with restriction being a down-sampling to a smaller grid and prolongation being an up-sampling (interpolation). The basic multigrid procedure is illustrated in Figure 10.4.

The precise details of the multigrid implementation, such as number of iterations performed on each grid and strategies for determining when to transfer between two grids only influences the time taken to converge. It does not alter the actual solution itself, and given that we are only really interest in multigrid as a tool to solve some equations we do not discuss these details here.

10.3.3 Biharmonic

We now turn to numerical solutions to the biharmonic, Equation 10.2. Solving in a similar way we first multiply it out giving:

$$\left(\frac{\partial^4}{\partial u^4} + 2\frac{\partial^4}{\partial u^2 \partial v^2} + \frac{\partial^4}{\partial v^4}\right) \underline{X}(u, v) = 0 \tag{10.24}$$

Recall Equations (10.16), (10.17) and (10.18). We now use these to derive an

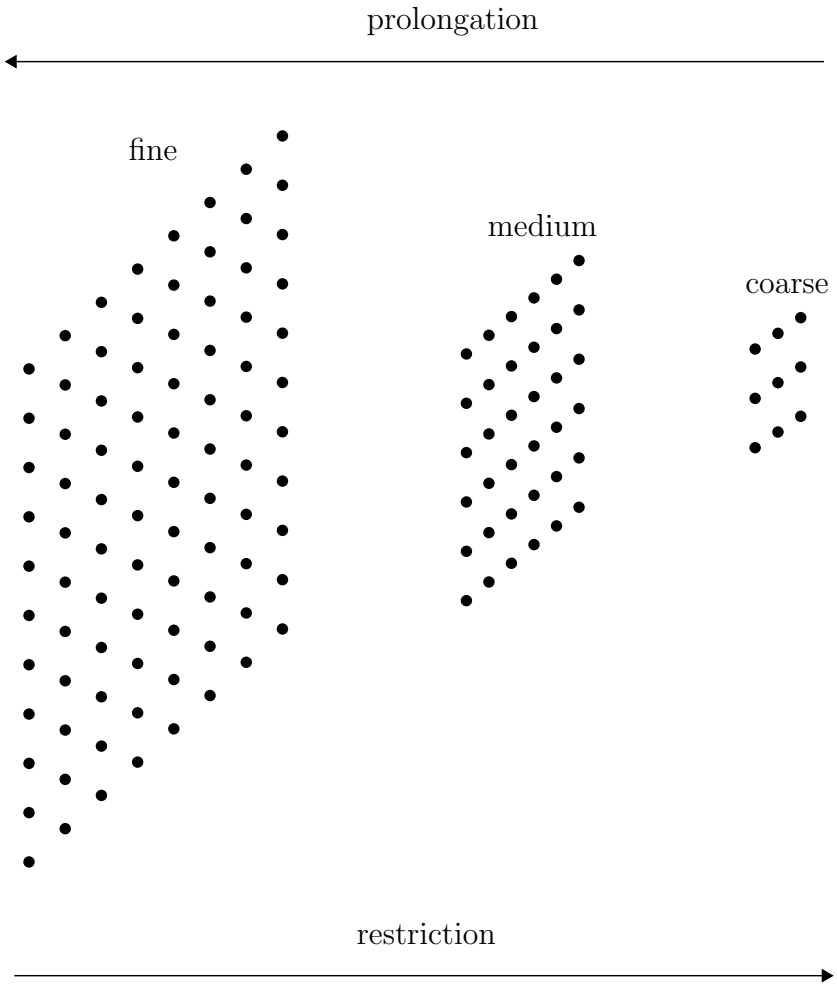


Figure 10.4: The three essential operators for multigrid solutions

approximation for $\frac{\partial^4}{\partial u^4}$, $\frac{\partial^4}{\partial v^4}$ and $\frac{\partial^4}{\partial u^2 \partial v^2}$, as required for Equation 10.24:

$$\frac{\partial^4}{\partial u^4} \approx \frac{1}{\Delta^4} \left[f(u+2\Delta, v) + f(u-2\Delta, v) - 4f(u+\Delta, v) - 4f(u-\Delta, v) + 6f(u, v) \right], \tag{10.25}$$

and similarly

$$\frac{\partial^4}{\partial v^4} \approx \frac{1}{\Delta^4} \left[f(u, v+2\Delta) + f(u, v-2\Delta) - 4f(u, v+\Delta) - 4f(u, v-\Delta) + 6f(u, v) \right]. \tag{10.26}$$

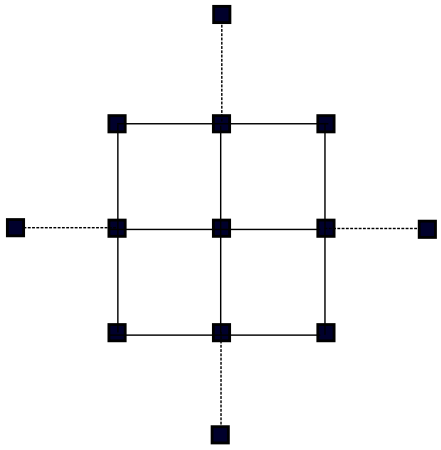


Figure 10.5: Visual illustration of the 4^{th} order 2-D scheme

Then

$$\begin{aligned}\frac{\partial^4}{\partial u^2 \partial v^2} \approx \tfrac{1}{\Delta 4} \quad & [f(u+\Delta, v+\Delta) - 2f(u+\Delta, v) + f(u+\Delta, v-\Delta) \\ & -2f(u, v+\Delta) + 4f(u, v) - 2f(u, v-\Delta) + f(u-\Delta, v+\Delta) \\ & -2f(u-\Delta, v) + f(u-\Delta, v-\Delta)]\,. \end{aligned} \tag{10.27}$$

Substituting Equations (10.25) through (10.27) into Equation 10.24 we get a complete discrete scheme (illustrated in Figure 10.5) for the biharmonic:

$$\begin{aligned}\frac{1}{\Delta^4} \quad [\quad & f(u+2\Delta, v) + f(u-2\Delta, v) - 8f(u+\Delta, v) - 8f(u-\Delta, v) + 20f(u, v) \\ & +f(u, v+2\Delta) + f(u, v-2\Delta) - 8f(u, v+\Delta) - 8f(u, v-\Delta) \\ & +2f(u+\Delta, v+\Delta) + 2f(u+\Delta, v-\Delta) + 2f(u-\Delta, v+\Delta) \\ & +2f(u-\Delta, v-\Delta)] = 0. \end{aligned} \tag{10.28}$$

Notice that in the case of the 4^{th} and higher order schemes, near the edges of our grid we will need to compute values for points outside of the domain itself in order to achieve the desired continuity right up to the boundary. This is because the higher order schemes express a given point in terms of more than just its

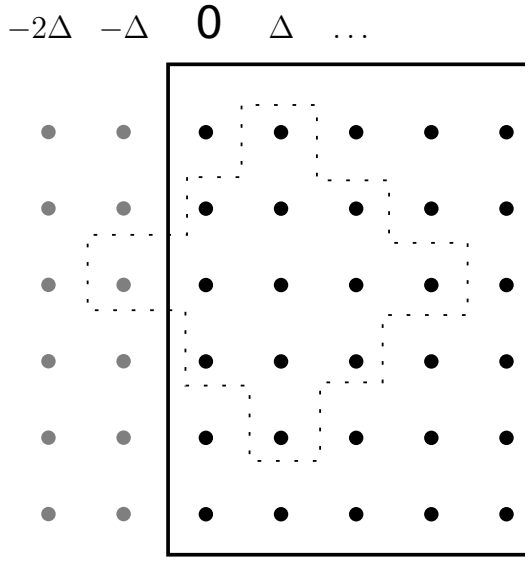


Figure 10.6: At the edges of the grid ghost points are calculated to fill in for the higher order schemes

immediate neighbours, yet we seek a solution to these equations right up to the edge of the domain where our positional boundary conditions have been defined. These points are often referred to as *ghost points*, illustrated in Figure 10.6. The ghost points are trivial to compute, using the boundary conditions directly, such that the solution would pass through the boundary.

10.4 Higher order elliptic PDEs

We can generalise the Laplacian (Equation 10.15) and Biharmonic (Equation 10.2) to any positive, integer order, o:

$$\left(\frac{\partial^2}{\partial u^2} + \alpha^2 \frac{\partial^2}{\partial v^2}\right)^o \underline{X}(u, v) = 0. \tag{10.29}$$

For both the analytic solution and the numerical one this only requires small extensions to the solutions outlined previously.

10.4.1 Analytic

In the case of the analytic solution we have to make several minor changes, for example for a 6^{th} order elliptic PDE (Kubiesa et al., 2004) Equation 10.4 remains the same, but the constants, Equation 10.5 becomes:

$$\begin{aligned}
\underline{A}_0 &= \underline{a}_{00} + \underline{a}_{01}u + \underline{a}_{02}u^2 + \underline{a}_{03}u^3 + \underline{a}_{04}u^4 + \underline{a}_{05}u^5, \\
\underline{A}_n &= \underline{a}_{n1}e^{\alpha nu} + \underline{a}_{n2}ue^{\alpha nu} + \underline{a}_{n3}u^2e^{\alpha nu} + \underline{a}_{n4}e^{-\alpha nu} + \underline{a}_{n5}ue^{-\alpha nu} + \underline{a}_{n6}u^2e^{-\alpha nu}, \\
\underline{B}_n &= \underline{b}_{n1}e^{\alpha nu} + \underline{b}_{n2}ue^{\alpha nu} + \underline{b}_{n3}u^2e^{\alpha nu} + \underline{b}_{n4}e^{-\alpha nu} + \underline{b}_{n5}ue^{-\alpha nu} + \underline{b}_{n6}u^2e^{-\alpha nu}.
\end{aligned} \tag{10.30}$$

Here we also need two extra, higher order, boundary conditions now as well and the matrix M which we used to calculate values for A_n and B_n is expanded to:

$$\begin{bmatrix}
1 & 0 & 0 & 1 & 0 & 0 \\
1e^{\alpha n} & e^{\alpha n} & e^{\alpha n} & 1e^{-\alpha n} & e^{-\alpha n} & e^{-\alpha n} \\
\alpha n & 1 & 0 & -\alpha n & 1 & 0 \\
\alpha n e^{\alpha n} & (1+\alpha n)e^{\alpha n} & (2+\alpha n)e^{\alpha n} & -\alpha n e^{-\alpha n} & (1-\alpha n)e^{-\alpha n} & (2-\alpha n)e^{-\alpha n} \\
(\alpha n)^2 & 2\alpha n & 2 & (\alpha n)^2 & -2\alpha n & 2 \\
(\alpha n)^2 e^{\alpha n} & (2\alpha n + (\alpha n)^2)e^{\alpha n} & (2+4\alpha n + (\alpha n)^2)e^{\alpha n} & (\alpha n)^2 e^{-\alpha n} & (-2\alpha n + (\alpha n)^2)e^{-\alpha n} & (2-4\alpha n + (\alpha n)^2)e^{-\alpha n}
\end{bmatrix}. \tag{10.31}$$

In a similar fashion we can solve for any value of o, provided boundary conditions are available.

10.4.2 Numerical

In order to solve higher order equations we have to derive new discrete terms. This can be formulated in terms of existing lower order terms, and is trivial to perform, both on paper and algorithmically. The resultant schemes become very unwieldy to present on paper however and do not add much to the discussion here, although since they are relevant to this thesis we have included them in Appendix G.1. One such scheme, for a 6^{th} order elliptic PDE, the same as used

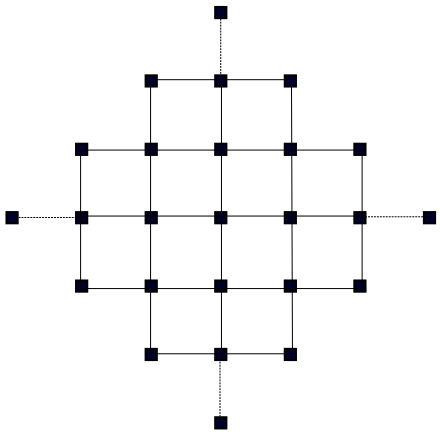


Figure 10.7: Visual illustration of the 6^{th} order 2-D scheme

previously for the analytic solution, is illustrated in Figure 10.7.

10.5 Higher dimension PDE surfaces

This section details our method to extend PDE surfaces to high dimensional spaces. We build upon both the analytic numerical solutions discussed earlier in this chapter. We show how both can be used straightforwardly to increase the dimension of the space the surface is embedded within. We then further develop just the numerical schemes of Section 10.3 for increasing the number of variables that parametrise the surface, since we cannot do this for the analytic solution.

10.5.1 Increasing the dimensionality of the embedding space (extrinsic dimensionality)

Given the solution outlined in Section 10.2 it is straightforward to increase the dimensionality of the space in which the surface is embedded. In the context of modelling image manifolds this corresponds to the resolution of the sample images used. To achieve this for the analytic solution one simply has to increase the dimensionality of the vector-valued functions $\underline{A}_0$, $\underline{A}_n$ and $\underline{B}_n$. This in turn simply requires one to specify boundary conditions for the PDE in each additional

dimension, and solve the corresponding linear system of equations.

The same principle applies to the method for increasing the dimensionality of the embedding space of the numerical solution. In this instance we now have to solve on one grid per dimension, and remove the assumption that there will be three.

In short since each dimension of the space in which the surface is embedded is treated individually (see Equation 10.1) by both solutions the number of these dimensions can be increased by adding appropriate boundary conditions and solving once per dimension instead of 3 times.

10.5.2 The parameter space (intrinsic dimensionality)

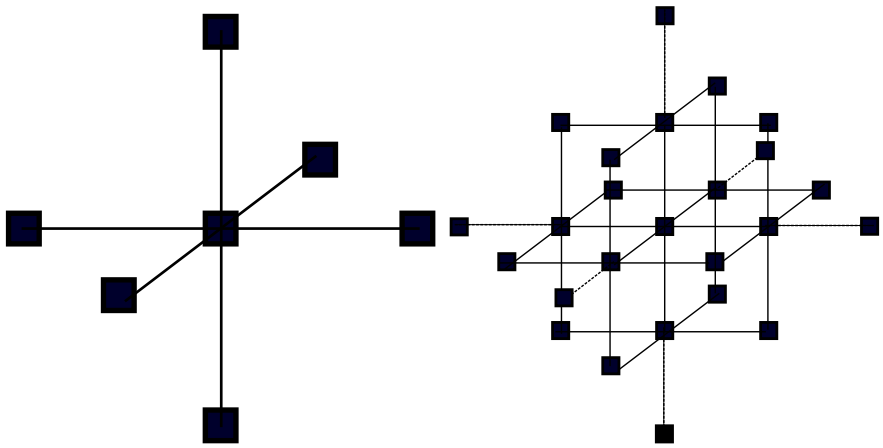
Extending the surface to be controlled by more than 2 parameters (an n-manifold) is a more involved problem — the analytic solution outlined in Section 10.2 is no longer applicable, because we cannot generalise Equation 10.4. We are now forced to turn exclusively to the numerical solution outlined in Section 10.3. The only existing work in the context of either graphics or vision that we are aware of is (Du and Qin, 2007), which has only addressed the three dimensional case, and only from the perspective of volume rendering (i.e. 3-D embedded in 3-D). The numerical scheme produced from our general solution is in the 3-D case however identical to their proposed scheme.

We firstly generalise the elliptic PDEs we have considered to this point to be:

$$\left(\sum_{n=1}^{N} \frac{\partial^2}{\partial x_n^2} \right)^o = 0, \tag{10.32}$$

for any N variable PDE.

Solving this is now an extension of what we have seen previously with the numeric solution. To derive a discrete scheme for solving this we now merely have to repeat the procedure we used to generate the discrete versions of the individual



(a) 2^{nd} order 3-D scheme (b) 4^{th} order 3-D scheme

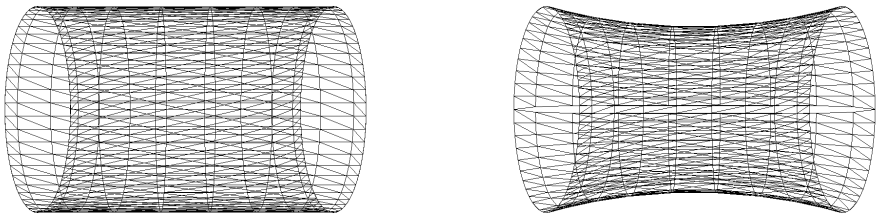
Figure 10.8: Visual illustrations of two schemes

terms of Equations (10.2) and (10.15). This involves creating terms for the $\frac{\partial}{\partial x_n}$, and using these to generate the remaining higher order terms. A number of three dimensional schemes are illustrated in Figure 10.8. Illustrations of schemes stop at the three dimensional case, however we have produced and run six dimensional schemes.

10.6 Implementation

For the purposes of testing, a simple example application, shown in Figure 10.9 was produced to demonstrate the two different solutions running side by side for the biharmonic (Equation 10.2). In both cases the boundary conditions specified were identical. The implementation used here utilises a simple multigrid solver, with symmetric Gauss-Seidel relaxations performed at each stage. The numerical solution (b) has a relatively high maximum residual error ($\epsilon_{\max} = 0.01$), but only required a small number of iterations. If we perform more iterations and continued until a smaller maximum residual error is obtained, then the numeric solution does indeed converge towards the analytic solution.

With this simple application a number of other tests were conducted to verify the implementations. Several of the results are presented here.



(a) Analytic (b) Numeric

Figure 10.9: Visual comparison of the results of the two different methods for the biharmonic PDE

Some example computation times with the schemes we produced are presented in Table 10.1. In all cases a 20^d grid was used. The final scheme failed to run because of memory requirements. In this test the number of iterations has been limited to 1000, as well as a maximum residual error being set. Note that in this table the 4^{th} order 3-D scheme we use is the same as in (Du and Qin, 2007). This implementation has not been particularly aggressively optimised yet, and in practice significant performance gains can most probably be made. Primarily this table shows us that the run times for any experiments with the numeric solution will increase dramatically as the order increases, as well as the dimension. The indication of this is that for a number of our larger, higher dimensional datasets we may be unable to complete all of the experiments.

In Figure 10.10 the computational cost, with the analytic solution, of increasing the dimension of the space in which the surface is embedded is investigated. As we would expect the time is proportional to the dimensionality of the space, which for models of image manifolds means that the runtime is indeed proportional to the resolution of the images.

Finally, in Figure 10.11 we investigate the time taken to converge, using the same setup as in Figure 10.9(b).

Table 10.1: Run times with the numerical schemes

Method/scheme	Computation time per dimension	Max Residual error (Iterations)
2^{nd} Order 2-D numeric	0.010s	0.00854 (13)
2^{nd} Order 3-D numeric	0.320s	0.00949 (15)
2^{nd} Order 4-D numeric	29.250s	0.0118 (17)
4^{th} Order 2-D numeric	0.020s	0.0098 (90)
4^{th} Order 3-D numeric	1.180s	0.00998 (137)
4^{th} Order 4-D numeric	197.430s	0.00985 (180)
6^{th} Order 2-D numeric	0.090s	0.00999 (544)
6^{th} Order 3-D numeric	7.030s	0.0118 (1000)
6^{th} Order 4-D numeric	1034.530s	0.0442 (1000)
6^{th} Order 6-D numeric	*dnf*	*dnf*

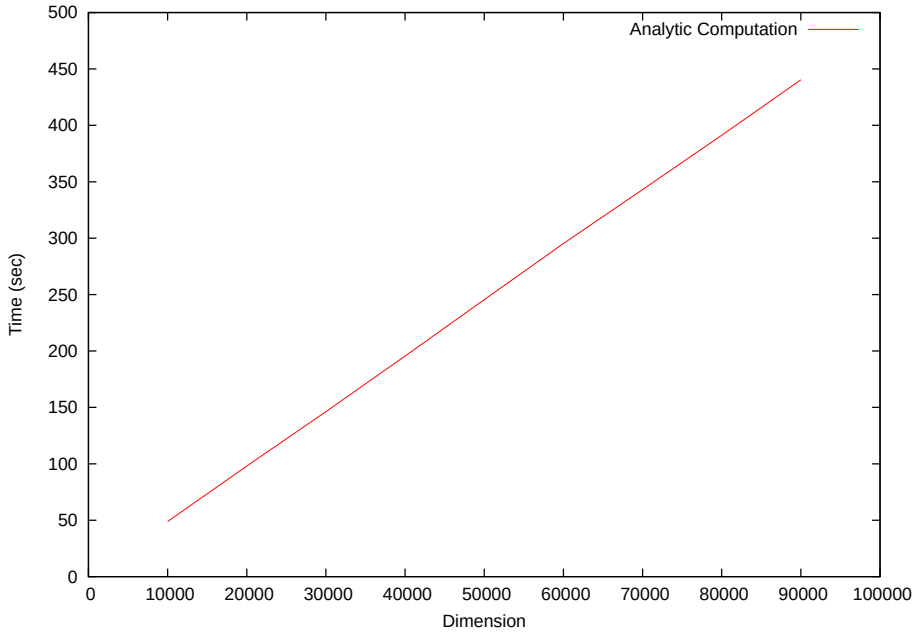


Figure 10.10: Computational results from analytic solution

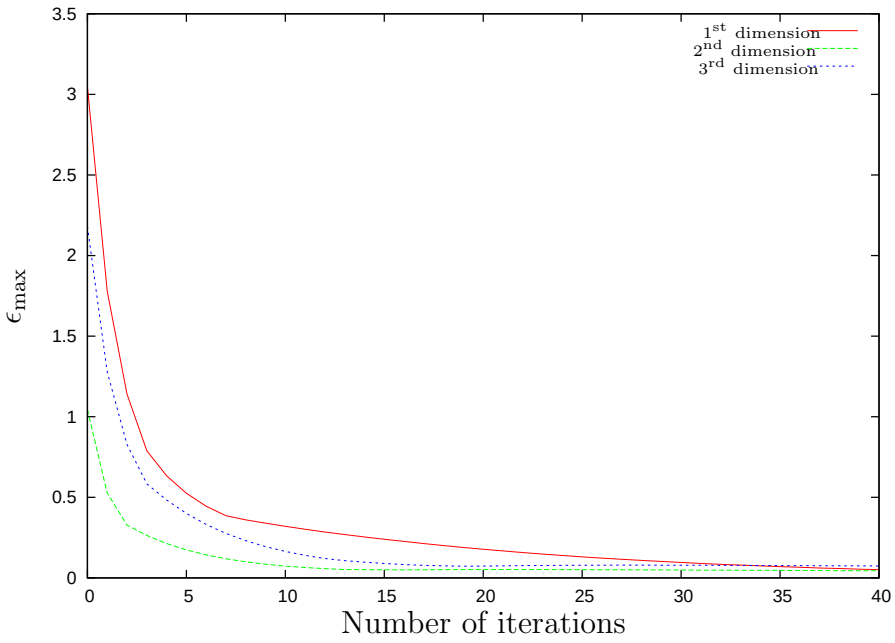


Figure 10.11: Maximum residual error vs. iterations with the biharmonic scheme

10.7 Experimental setup

As with the previous two models we investigate everything at a number of different (gap) scales. We use the same set of scales that we previously considered with the NURBS and linear models in order to facilitate future comparisons.

The first and most basic point to note is that we only have direct control over the position of any PDE surface at the boundary conditions, i.e. either $u = 0$ or $u = 1$. This implies that we are going to need to manage sub-division of the whole manifold (i.e. breaking it down into small segments and *blending* them) ourselves.

In common with our NURBS model the PDE model introduces a number of additional factors to consider above and beyond the scale (gap) when running our models, besides just the spacing between the samples. With the analytic solution we have the following factors to consider:

- Number of Fourier modes to use (1, 2, 6, 10, 20, 50)
- Use of a remainder term (Yes/No)
- Value of α (-1, 0, 0.5, 1, 2, 10)

- Order of the equation used (2, 4, 6)

For the numeric solution we have the following list instead:

- Size (i.e. resolution) of the grid (10, 20, 50)
- Value for $\epsilon_{\max}$ (0.1, 0.01, 0.0001)
- Order of the equation used (2, 4, 6)

With each of these parameters to consider the values used for the parameter search were heavily restricted in order to produce a small enough set of experiments, so as to be feasible to run to completion. The parameters chosen are somewhat arbitrary, but based on initial testing using the prototype application we constructed and artificial data. The order of the equation used for instance, is also based upon the literature surveyed, with most authors using only 4. For the number of Fourier modes used, 6 is typically reported in the literature, 1 and 2 are extremely low (any lower and by definition we would not have been doing Fourier analysis), whilst 50 is extremely large. The configurations we generated are listed in Appendix G.

A number of our datasets are not suitable for use with the PDE model. None of the one dimensional datasets (mostly from panoramic camera on robots) are applicable here. Furthermore with the three or more dimensional datasets we can only run the numerical model and not the analytic one. We could have decided to use slices of these datasets here at this point, however these would not be comparable to other models, at least not without doing the same for all the other models. Doing this for every possible slice of every dataset would further increase the CPU time required[2], and further increase the difficulties inherent in generating meaningful summaries of results. We can perfectly adequately test the PDE surface method as a manifold model from the subset of our data that is

[2]The results presented in this thesis has already used in excess of 100,000 hours of CPU time!

compatible, without doing so. Our conclusions for the idea of modelling image manifolds as a whole will be based upon all of the results, and could have been based on just one suitable model.

There are also two important issues to be addressed when using PDE surfaces to model image manifolds. Firstly we stated right at the beginning that we were assuming the surface was periodic in v. The observant reader may have already noticed this is the case in a number of our datasets. This was not by accident either[3]. The 2DWOODBOX, 2DTEAPOT, CHESSBOARD and all of the KNIGHT_ datasets are naturally periodic in v. We will naturally do the sensible thing for these datasets and ensure that v is indeed mapped onto the periodic part of the surface.

For some of the other datasets, such as EXPT03 and WINDOW3 there is no obvious periodicity in the sampling. The question then becomes one of how we should handle this. The usual strategy in the PDE surface literature is to arbitrarily connect (i.e. close) one dimension. It would probably make sense to pick the one which is closest to being already connected, but as we have already discussed the concept of 'closeness' is strongly tied to the concept of a metric (Chapter 5), and therefore quite open-ended. Furthermore in early stage testing it made almost no observable difference. In essence what we do to address this problem is to pretend to the solver that the input is periodic in v and then ensure that in the output from the solver we filter out anything that happens as a result of this. An example of this is illustrated in Figure 10.12. In this example we would make the dataset periodic, by implicitly connecting the top row to the bottom row, and re-mapping the output so as to discard any images generated as a result of this.

One further point to note with regards to the periodicity of the data — the numeric solver could be modified to accept entirely non-periodic inputs, but this would introduce two further problems. Firstly we would no longer be solving the

[3]It didn't make sense to point this out when we introduced the datasets though.

PDE we claim to be solving. Secondly we would need to specify some 'ghost points' at these new edges on the grid. Whilst this would, at least theoretically be possible, by removing the periodicity from the system we introduce a second set of boundaries, for which we do not have the boundary conditions required to generate ghost points from.

The second important issue for us to consider when building models of image manifolds using PDE surfaces is what we use for the boundary conditions. For our PDE surface we need to explicitly specify both positional, and at least first order derivatives at either end of the sub-section of our image manifold that we use as a PDE surface.

At the extremes of our manifold we need some way of estimating (or ideally calculating) the derivatives of the surface. In practice the most sensible way to achieve this (for 1^{st} order derivatives at least) is to disregard the outer set of points and use them instead to calculate a vector field at the second and penultimate set of points. We can then use this vector field as an estimate of the derivatives at the edges of the remaining data.

For derivatives within the rest of the manifold we can ensure that the sub-sections of the overall surface are 'blended', by using a similar process to what we used to calculated the derivatives at the end to calculate derivatives throughout the interior. In this way derivatives at the end of one sub-section of the surface will match the derivatives at the beginning of the next.

This whole process is illustrated in Figure 10.12, where a 'gap' of 2 has been used. The input has been made periodic in v. Additionally this illustrates the use of the first and final set of points to calculate the 1^{st} order derivatives for the boundary conditions. This of course means that we will not be able to generate any output points for either of these columns, or the light grey ones which represent what would normally be the unseen points. The s_n's represent the n sub-sections of surface we have to build to represent the whole manifold, using all the input

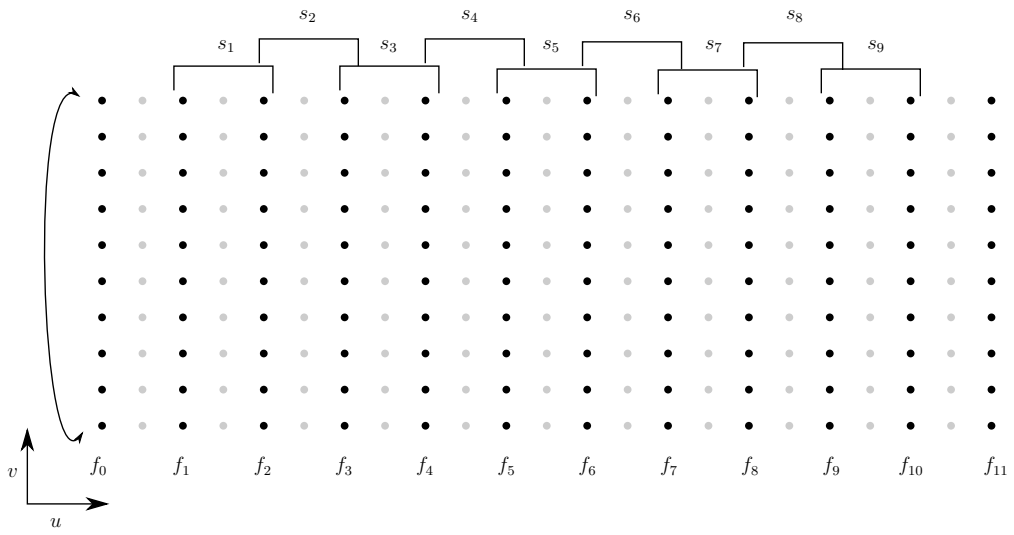


Figure 10.12: Illustration of the experimental setup of the PDE surface model

sample points. The s_n use f_n and f_{n+1} as the positional boundary conditions, for the boundary conditions $f_{n-1} - f_n$, $f_{n+2} - f_{n+1}$ as the 1^{st} derivative boundary condition. The s_n do not overlap at all, they exactly meet at each f_n.

Finally we should note that when a point is requested from the numerical model, which does not exactly match a point in the discrete domain the nearest available point is returned instead. That is to say our novel images are selected and synthesised during model building (i.e. constructing the discrete domain) instead of later on when requested.

10.8 Results

In the results there are a number of observations we would expect to make. Firstly there are quite a few cases where we would expect to observe that, even though the output parameters exactly match a known input point the image generated does not match. This will occur with the analytic solution when we truncate the Fourier series early. Without the remainder term this will undoubtedly occur, but even in the case where we have used a remainder term if the series has been truncated prematurely the remainder term could still poorly approximate what is

left, so we will most likely see this a number of times in our results.

We would also expect that the number of iterations, or the error tolerance we allow in the numeric solution will influence the observed results notably. In all cases we will be expecting results that look 'smooth' in PCA plots. Anything to the contrary is likely to indicate a problem.

We also know the α parameter will influence the interior of each segment of surface we generate, and thus the images generated. However what influence this is likely to have is somewhat less obvious at this stage.

Whilst what we have performed here is effectively a brute force parameter space search it is important to note that the samples we have as a result of this search are not as regular and complete as it may have appeared from the initial description of the parameters and values used. There are a number of reasons why this is the case. Firstly we produced 1700 configurations from these parameter values, which is 1700 experiments per dataset. A number of these configurations require prohibitively large amounts of memory to run, and the experiments were run on an inhomegenous cluster. Whilst every effort was made to predict the memory requirements of each job for each dataset and schedule things suitably this process was 'best-effort' and not perfect, which resulted in a number of experiments being missing for some or all datasets. In another number of cases we either produced configurations which were simply unworkable — for instance with the analytic solution it is possible to produce systems of equations which were not as trivially solvable as our implementation assumed. In these cases an error was logged and no result returned. Several configurations produced results unacceptably slowly, an upper limit of 10,000 minutes was placed on the run time for any experiment to avoid long running configurations dominating all the available resources.

The results themselves are presented in full in Appendix G.

10.9 Discussion

As with the previous two chapters, our discussion here will focus solely on issues that affect only the PDE surface results. We are deferring 'global' discussion until the next chapter. There are quite a number of interesting points to consider at this stage still. Primarily though, given that we have effectively performed a parameter space sweep here with the construction of our models, the results of this are of interest. Additionally we are interested in a practical comparison of the two different solvers we implemented, beyond the obvious difference discussed earlier in this chapter.

In order to analyse the impact of varying just one of the parameters we considered we shall be presenting a number of examples, where the remaining other parameters have been fixed for detailed inspection. This has the unfortunate consequence however that what we show could potentially be only representative of the effect of varying the parameter in question in the specific configuration considered. We address this problem in the simplest way possible here; by including and comparing results from a number of different configurations we can begin to see if our observations are representative of the results as a whole, or merely an artefact of the configuration in question. We deliberately exclude from this process the manifolds we have introduced which we know are unlikely to perform well, such as Faces and BMnoise. Throughout all of the graphs in this section Sinecos uses the left-hand y-axis and the other manifolds use the right-hand one, to allow sensible scales to be shown for the Sinecos manifold.

We have divided the analysis of these results based on the two different solutions we have formulated for the PDE surface model. For the purposes of these discussions, configurations and datasets where complete or mostly complete results were available have been selected. In cases where one or more results were still missing experiments were manually rescheduled for the purposes of this discussion. In all instances within this discussion we consider our results using the

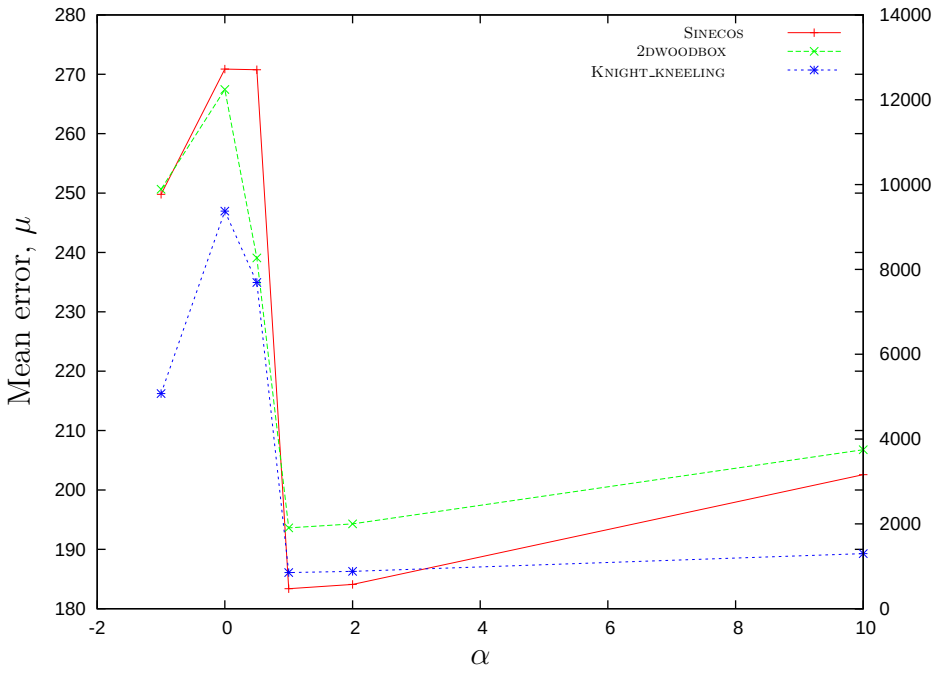


Figure 10.13: The effect of varying α (4^{th} order, $N = 6$, with remainder term)

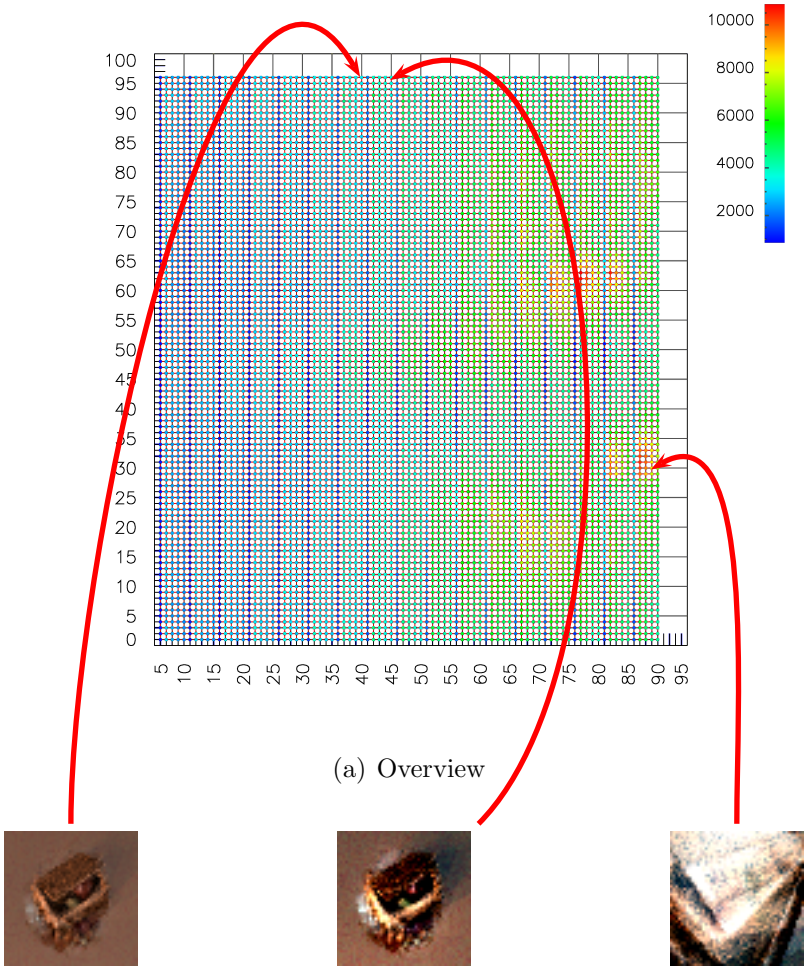
Euclidean distance, which is considered the 'usual' distance metric.

10.9.1 Analytic

For the analytic solution we only have results from the two dimensional image manifolds to consider, given that the solution is not applicable to any other manifolds. There are a number of parameters to this solution though, which we shall address in turn here.

10.9.1.1 α value

The influence of the value of the α parameter is relatively straight forward to see, and understand. Figure 10.13 shows the effect of varying this parameter, when the other parameters are fixed. In this instance there appears to be a minimum around $\alpha = 1$. This makes sense given that in the CAD style PDE surface method applications the value of the α parameter influences the shape of the surface by either 'pinching' or 'inflating'. For the case of image manifold modelling neither of these two shapes bear any resemblance to anything we have observed in image



(a) Overview

(b) Detail at the boundary, $(41, 95)$

(c) Detail in the interior, $(44, 95)$

(d) Detail in the interior, $(88, 30)$

Figure 10.14: 2DWOODBOX, $\alpha = 10$

space. For this reason therefore it is not unsurprising to find that a value of $\alpha = 1$ which would produce a cylinder in the 3-D case is close to, if not exactly, the minimum.

This explanation is further supported by more detailed inspection of the results. For example, in Figure 10.14 we show a detailed plot, in the parameter space of the manifold to avoid clutter. This plot shows the error reported by the Euclidean distance metric for $\alpha = 10$ with the 2DWOODBOX dataset. In this plot we can see a number of features which support what we have suggested about the value of α as it moves away from 1. Firstly, in the overall view 'banding' is significantly visible, by which we mean there are observable vertical stripes of low error which

exactly match the points where $u = 0$ and $u = 1$, i.e. the boundary conditions. This is to be expected — for any value of α the surface always passes through the imposed boundary conditions, it is between the boundaries that the choice of α influences the shape of the surface. When we pick any two pairs of images, ((b), (c)) such that one falls on a boundary condition, and the other falls as far from a boundary as possible we can inspect the artefacts in the image. Primarily we can observe that, compared to the results from the linear model (Chapter 8) the distortions in the generated image are not identical. We seem to have 'lost' brightness in the image, which is one of the effects of a poor choice of α. In other examples (e.g. (d)) the exaggeration to the shape of the surface generated caused by the large α value causes the opposite effect, a net increase in brightness of the generated image. Both the under and over brightness in these images is caused by the choice of α pulling the surface outside of the $[0, 255]$ range of the images, and therefore the generated image being clamped.

10.9.1.2 Number of Fourier modes

We now turn our attention to the impact of the number of Fourier modes used to construct the surface. We have deliberately avoided using the remainder term here, since the remainder term can potentially mask many of the problems that can be caused by truncating the Fourier series.

In Figure 10.15 we show the typical effect of increasing the number of Fourier modes used. We know, by definition, that the SINCECOS manifold would be directly representable as a Fourier series, provided we avoid aliasing problems, which are the dominant feature in it. This accounts for the peak at $N = 20$ with the SINECOS manifold which was not observed with the other manifolds.

With the 2DWOODBOX dataset we can observe a minimum around the $N = 6$ area. The number of points used in this graph is insufficient to determine exactly where this minimum occurs, however there are a number of interesting points to

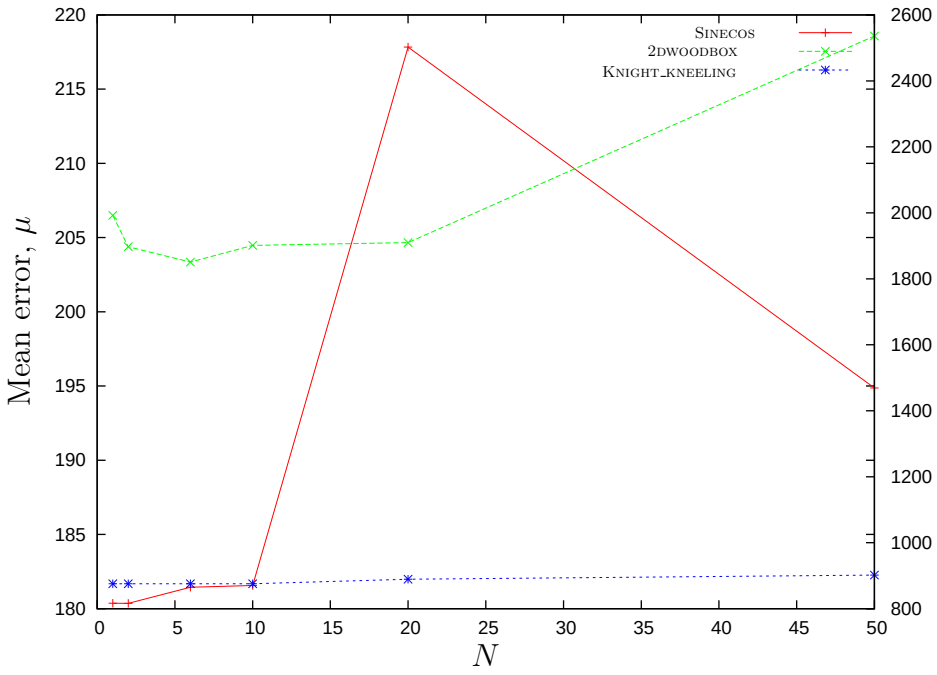


Figure 10.15: The effect of varying the number of Fourier modes used (others params)

draw still. As would be expected, a low number of Fourier modes causes an increase in the error. This is to be expected really given that truncating the series early, and not introducing a remainder term amounts to throwing away high frequency information.

Interestingly though increasing the number of Fourier modes beyond $N = 6$ causes the error to rise again. The reasons for this are most likely at least two fold. Firstly, as we saw in our discussion on image metrics (Figure 5.3, page 89) it is easy to construct examples whereby loosing detail causes the distance reported by a metric to fall rather than rise. This is likely to be part of the cause of the increase in error as the number of Fourier modes used increases. Secondly though, at $N = 50$ (and even earlier) we start to encounter some numerical problems with the implementation. One example of the type of problems observed is shown in Figure 10.16, where banding, and colour errors are visible. It is unknown at this point exactly where the problem lies (the FFT itself, or within the PDE surface implementation), however this does seem to be the problem.

Figure 10.16: Errors in an image produced with $N = 50$

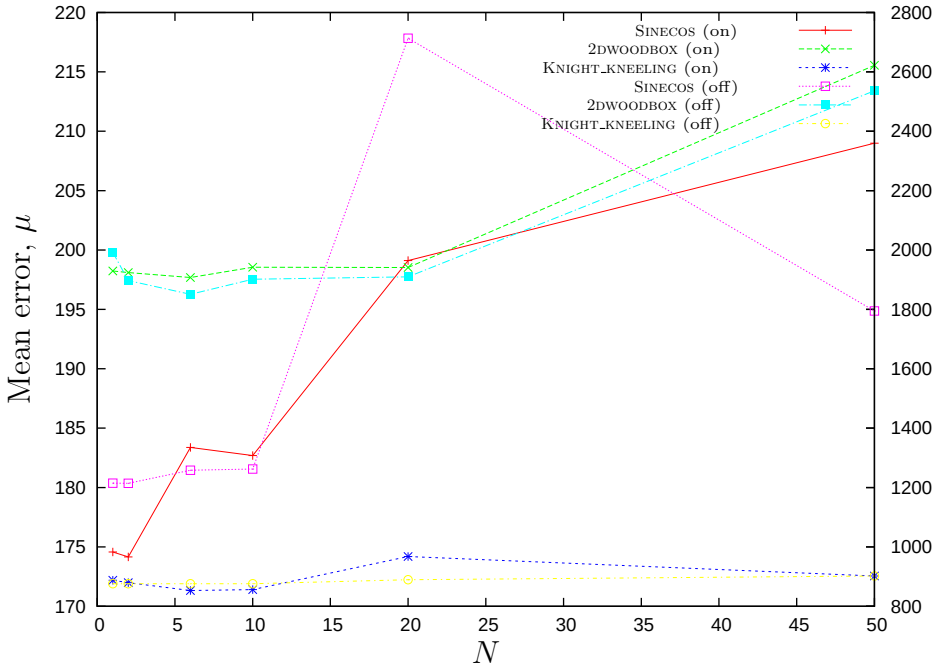


Figure 10.17: The effect of the remainder term

10.9.1.3 Use of remainder term

The remainder term is a boolean value — we either enable it, or we disable it. This means that plotting it on the x-axis of a graph as we have done with the other parameters we have discussed makes little or no sense. In Figure 10.17 we therefore repeat the plot in Figure 10.15, with the remainder term enabled. The earlier results are shown along side for direct comparisson.

The results from this test (with the exception of Sinecos which is a very contrived example) seem to suggest that the remainder term is not particularly useful for modelling image manifolds. This is not really as shocking as it might seem at first. With low numbers of Fourier modes (i.e. less than 5) the remainder term does help to improve the resulting images at the known boundary conditions, as would be expected. However the remainder term also apparently serves to

	With remainder	Without remainder
$g = 2, N = 2$	$\mu = 1421.77$	$\mu = 1755.79$
$g = 5, N = 2$	$\mu = 1923.94$	$\mu = 1897.24$

Table 10.2: Remainder terms at different size gaps, 2DWOODBOX

worsen the results in the interior stretches. Essentially therefore what determines the effectiveness of the remainder term with low (i.e. where precision is not a problem) values of N is the width of the gap parameter, which determines the ratio between the number output images that correspond to an input point to the number of previously unseen output images.

This can be confirmed by two further points. Firstly when $g = 2$, i.e. two thirds of the output images lie on a boundary the results from the use of the remainder term is much closer to the results without the remainder term in the cases where it is worse and more cases show an improvement. That is to say when more than half of the images fall on a boundary rather than between them the positive impact of improving the results on the boundary outweighs any negative impact it may have on the interior of the surface. Note that the average is only slightly better overall in every case because lying on a boundary is not the same as corresponding with an input image, for a 3×3 grid 4 of the points correspond to input points, whilst 5 of them do not, thus it is still not a majority of the points that match ever. Table 10.2 shows one example of this.

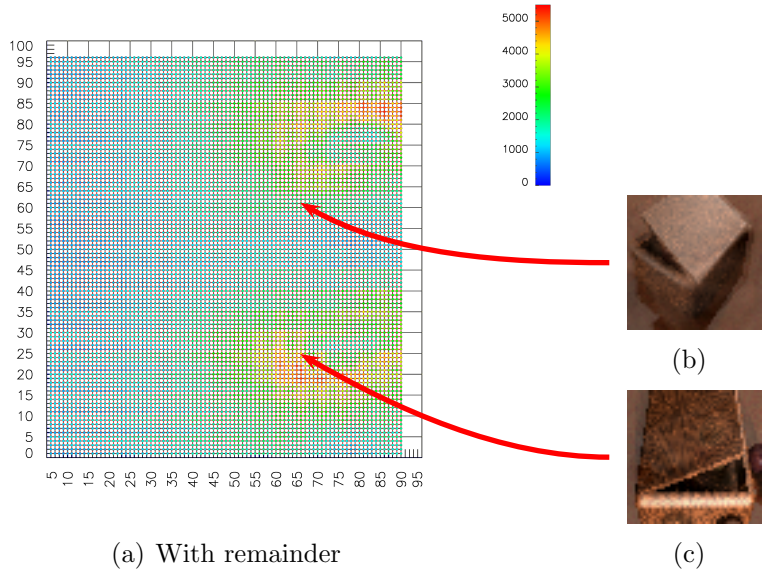
Secondly the inspection of Figure 10.18 shows us a number of interesting points about our solution to the PDE and our model itself. When the remainder term is used the resulting images are sharper and clearer, and around the areas where the surface matches up to an input image this results in improved output images and a corresponding reduction in the measured difference. Where the surface does not match exactly with an input image, i.e. it is on the interior of a gap, however, the story is quite different. The image is still sharper, but of course by not being correct to begin with that results in a greater measured error. In this instance the blurred effect of the image where no remainder term was used works as an

advantage in terms of the metric sometimes, in the same way that we saw earlier in Figure 5.3. This explains why (b) and (c) both look better than (e) and (f), yet (b) is considered to be a better output and (c) worse. Note also that the apparent disparity between (c) and (i) is not an error — the camera is moving towards the box fast at this point. The 'nearest neighbour' approach used by the application of the remainder term causes the resultant image to more resemble the closest sample point that was used in creating the remainder term than the actual, previously unseen ground truth.

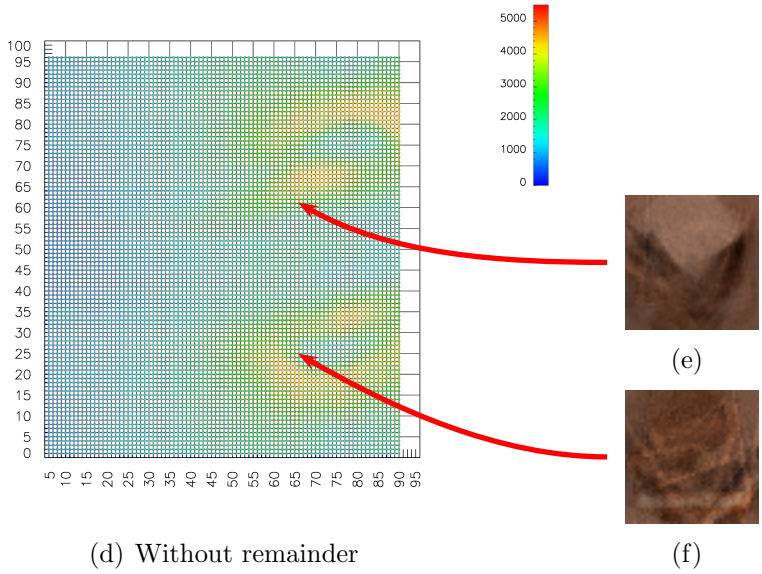
10.9.1.4 Order

When it comes to considering the order of the PDE used, Figure 10.19 shows that in all cases as we increase the order of the PDE used to model the surface the error increases. This is borne out by the full results, as well as just the cases illustrated here. This would seem to confirm that by blindly fitting higher order functions to our data we are in most cases forcing the surface in question to take a convoluted path in image space that does not match what the underlying data does. Note that the 6^{th} order result for the KNIGHT_FIGHTING manifold is missing because there were not enough sample points available to generate all the boundary conditions for it to run.

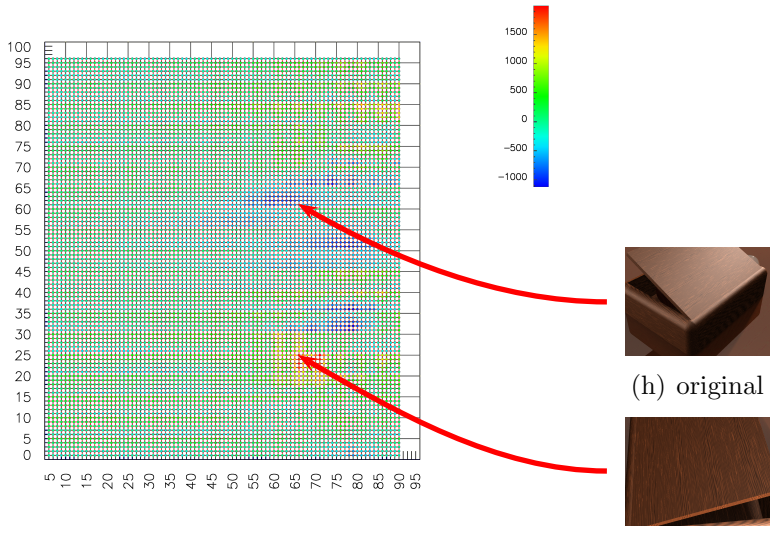
We are currently using the difference between two sets of images as an estimate of the derivative of the manifold for the construction of our boundary value problems, however this is far from ideal. Primarily this is because there is absolutely no guarantee that this estimate is in any way reflective of what the derivatives really are locally.



(a) With remainder (b) (c)



(d) Without remainder (e) (f)



(g) Difference (value is remainder - without remainder) (h) original (i) original

Figure 10.18: 2DWOODBOX, testing the remainder term, $N = 2$, $g = 5$

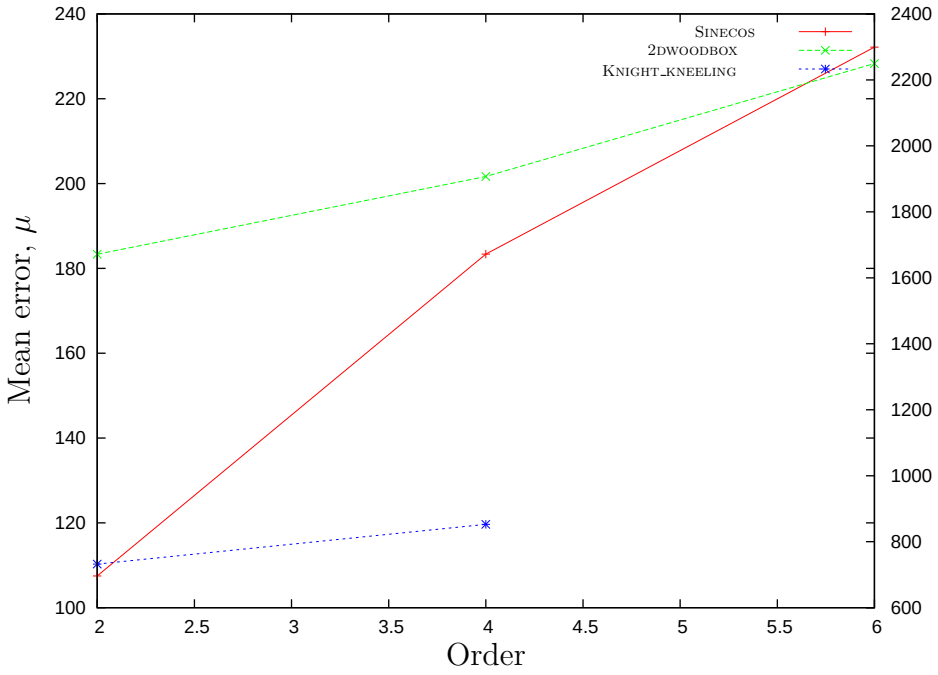


Figure 10.19: The effect of varying the order of the PDE used (analytic))

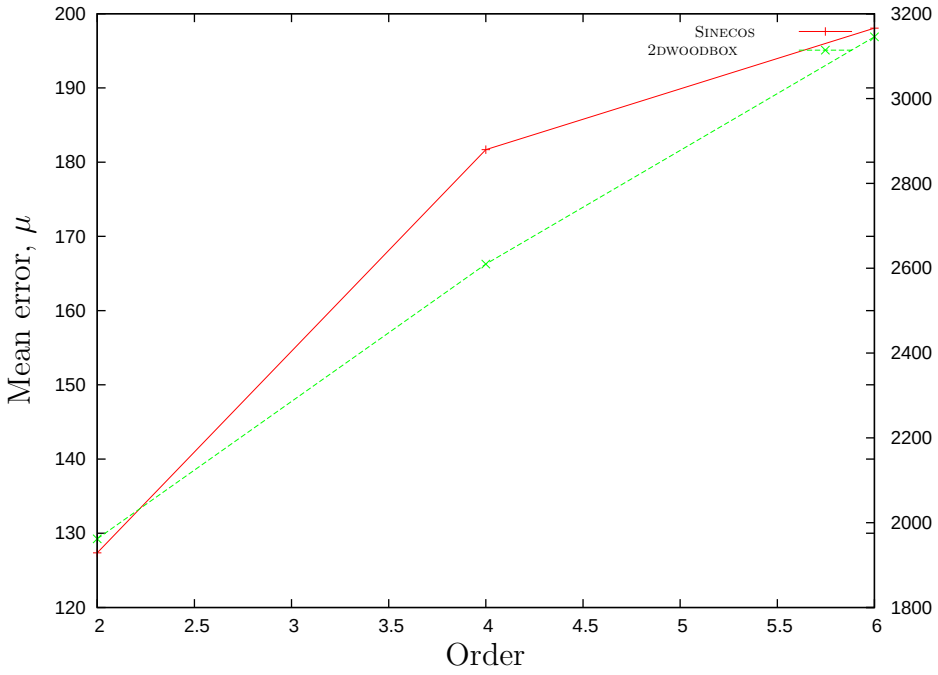


Figure 10.20: The effect of varying the order of the PDE used (numeric)

10.9.2 Numeric

10.9.2.1 Order

The investigation into the order with the numeric solution (Figure 10.20) to the PDE unsurprisingly yielded similar trends to the corresponding investigation for the analytic solution. This is because we are after all solving the same equations, just using an alternative method. The results can be worse than the equivalent analytic one however, possibly indicating that we have not solved the PDE as well as previously. Where they are better it often tends to be because the 'nearest neighbour on the grid' strategy we used returns a result the metric measures as 'better' than what our other solution would have returned instead.

In the (very few) cases where the results from the numerical solution have lower measured error than the analytic this is typically a manifestation of what Figure 10.9 shows, except in the case of higher order PDEs the 'sagging' could potentially counteract some of the contortions which the higher order surface construction has produced.

10.9.2.2 Grid resolution

One important point to note when considering the trend shown in Figure 10.21 is that the grid resolution will only be varying in the u direction. In the v direction the grid resolution is determined by the available boundary conditions[4]; we have deliberately avoided performing any interpolations that would be required to map discretely sampled boundary conditions of one resolution onto a discrete grid of another, different resolution.

It is perhaps slightly surprising to see that as we increase the resolution of the grid the error increases also. This can be explained largely by the fact that the increase in resolution will result in a better, smoother solution to the underlying PDE it represents. Of course if the PDE itself poorly represents the underlying

[4]That effectively means that the interpolation is only happening in the u direction!

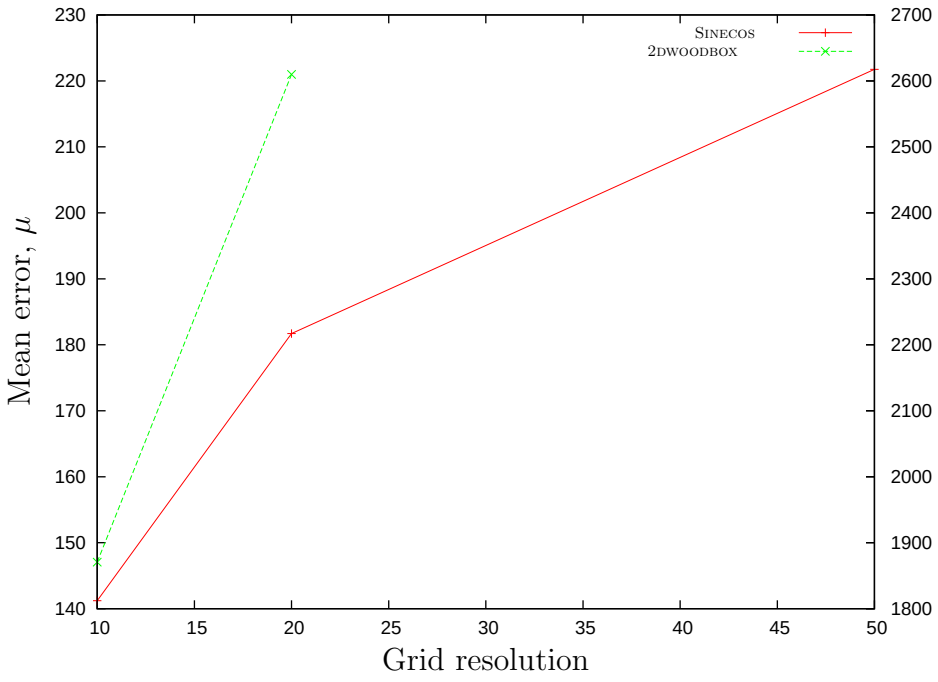


Figure 10.21: The effect of varying the resolution of Ω

surface we are trying to model there is no reason to assume that solving it better would actually reduce the measured error in the generated images, and indeed these results seem to confirm this. This in turn suggests that the PDE surface constructed does not reflect the underlying data well and furthermore, that by solving it better (i.e. enforcing the continuity requirements better) we only manage to move the solution further from where the data really is.

10.9.2.3 The maximum residual error ($\epsilon_{\max}$)

Initially it was planned to include an investigation into the choice of $\epsilon_{\max}$ at this juncture, but it was discovered that the results for it were being affected by the cap on the number of iterations performed, as well as the choice of $\epsilon_{\max}$. For example with $\epsilon_{\max} = 10^{-4}$ and the 2dwoodbox manifold for the 2^{nd} order PDE it typically required about 400 iterations per dimension, however for the 4^{th} and 6^{th} order most dimensions were hitting the 1000 iteration limit without seeing a suitably low $\epsilon_{\max}$.

The impact of this on the results table is likely to be minimal at worst however

— the lower order PDEs converge quickly (i.e. under the limit of iterations) and given that we are solving the same equations both numerically and analytically it would be most unlikely for the higher order numeric solutions to beat the lower ones, even if they were allowed longer to converge.

10.9.3 Comparative

We now introduce a further tabulated summary of results. In Tables 10.3 and 10.4 we directly compare the results from our two models. In these two tables 'missa' means both results for this entry are missing, 'miss1' means the analytic solution is missing and 'miss2' means the numeric solution is missing. In these tables a negative value indicates the analytic solution outperformed the numerical one.

Originally we planned to run the full set of configurations we outlined for the numeric model in Section 10.7, however in practice this proved to be too slow to be possible, so the results presented here are to be considered as 'proof of concept' rather than a complete run.

Table 10.3: Results from PDE analytic compared to PDE numeric, taxi

results of PDE analytic compared to PDE numeric (taxi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DWOODBOX	miss2	-8702.10	miss2	miss2	missa	miss2	missa
SINECOS	-6.07	-23.07	-22.05	66.78	missa	missa	missa

Table 10.4: Results from PDE analytic compared to PDE numeric, pdiff

results of PDE analytic compared to PDE numeric (pdiff)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DWOODBOX	miss2	-63.93	miss2	miss2	missa	miss2	missa
SINECOS	-0.22	-0.59	-0.40	-0.10	missa	missa	missa


(a) Analytic


(b) Numeric


(c) Difference image

Figure 10.22: Comparing an example from the two solutions (interior)


(a) Analytic


(b) Numeric


(c) Difference image

Figure 10.23: Comparing an example from the two solutions (boundary)

Given that for $g = 5$ with 2DWOODBOX the error from the analytic solution is 93731, a difference of 8702 between the analytic solution and the numeric one does not seem too large. Figure 10.22 shows two example images, from the interior of a surface. A number of artifacts are visible in the image from the numeric solution, from where individual pixels failed to converge within the maximum number of iterations permitted.

When we inspect images that fall exactly on a boundary (i.e. correspond to an input point) the two solutions are even more similar. Figure 10.23 shows one such example, where almost no visible differences are discernable between the two images.

The numerical solution was only introduced to work around the limitations of the analytic solution which cannot be used when there are more than two parameters for the manifold. In practice we have focused our experiments with the numeric solution on several specific areas, where we are interested in specific results instead of performing a global parameter search as we did for the previous two models. This has meant that we have focused on the 2-D manifolds, where the two solutions could be compared. In our testing of it we have seen, for all the cases we tried that the results are comparable and feasible, if a little slow to obtain. For most of the results the numeric solution performed worse than the

corresponding analytic solution, with the exception of SINECOS with large gaps where aliasing problems appear and influence the results.

10.10 Conclusion

This chapter has made two main contributions.

Firstly, the PDE surface method has shown itself to be a reasonable candidate for modelling image manifolds. The results from the analytic solution are fast to compute, and the images look somewhat plausible from surfaces built with a sensible configuration. The use of PDE surfaces in higher dimensional spaces (Woodland et al., 2007), particularly in image space, is a new contribution, and one which we believe can be expanded upon and improved in the future.

Secondly, the numeric solution we have introduced demonstrates, albeit somewhat slowly, that we need not be constrained to two dimensional datasets. This solution we have proposed, above and beyond the three dimensional case at least, is novel to the field of computer graphics as far as we are aware.

10.11 Summary

In this chapter we have seen:

- an introduction to PDE surfaces and two methods for solving them;
- how PDE surfaces can work in higher dimensional spaces;
- how PDE surfaces can work with general case n-manifolds;
- an application of this to the concept of image manifolds;
- results of this applied to our selected image manifolds;
- a discussion of these results.

In the next chapter we will explore:

- the results from all three methods, side by side;
- the limitations of our methods and future directions;
- what these results mean in the context of our original hypothesis.

Part IV

‘Bringing it all together’

Chapter 11

Comparing the three models

Having previously seen and discussed each of the three models individually for our datasets, we now look at the bigger picture, and address some of the issues these raised. Furthermore we are now in a position to answer questions like 'which of these three methods performed best?', and also discuss any conclusions we might draw from this. Specifically we are interested in finding instances where our higher order models outperformed the linear model.

We split our discussions such that initially we consider each possible pairing of our models in turn. In each instance we will pick out examples which are indicative of the general trend, as well as examples which run against the trend. As with previous chapters we consider two metrics here, namely taxi and pdiff. Full results for the rest of the metrics are once again available in Appendix H. In a number of instances no comparison could be made. As in Chapter 10, this is identified as one of three possible reasons using 'missa' to indicate both were missing, 'miss1' to indicate the first named model is missing and 'miss2' to indicate the second named model is missing. The value shown in the table itself indicates which of the two named methods had the lowest error, with a negative value indicating the first named method was lowest and a positive indicating the second named method was the lowest. Results for other metrics are shown in Appendix H.

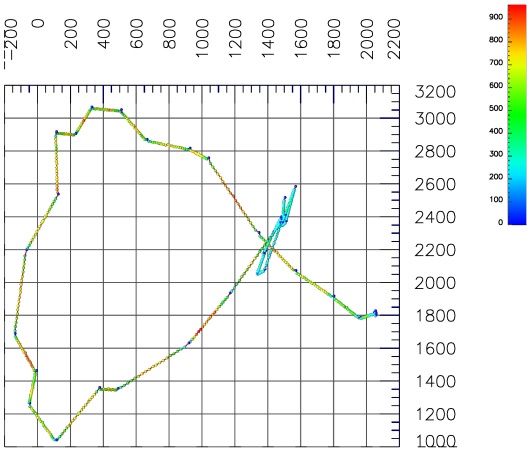
11.1 Linear vs. NURBS

Overwhelmingly the results in Table 11.1 and Table 11.2 show that the results from the NURBS model were outperformed by the corresponding results from the Linear model. Figure 11.1 shows one example comparison between the Linear and NURBS models. See also Figure D.13 for the corresponding raw PCA plot.

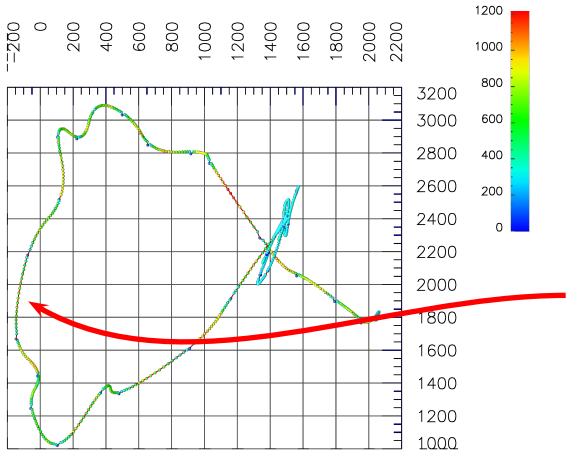
The first, possibly most notable difference between the two models, is the apparent smoothness of the generated manifold model in the NURBS cases, especially compared to the Linear model. This is to be expected of course — it was part of the aim of introducing the NURBS model and anything else would have indicated a failure of the model.

Initial inspection also suggests that the NURBS model does not offer any improvements over the Linear model. This is especially true if we consider the maximum error, as indicated by the scales associated with the colour maps. When we compare the two results, by subtracting the error in the Linear model from the NURBS one for each point we can see some interesting details however. One such example of this is shown in Figure 11.1(c). In this example we have adjusted the range of the colour map to only show values which are positive, i.e. where the NURBS model did outperform the linear model. The detailed stretch shown in Figure 11.1(c) has both the largest single value, as well as being the longest continuous stretch in this example.

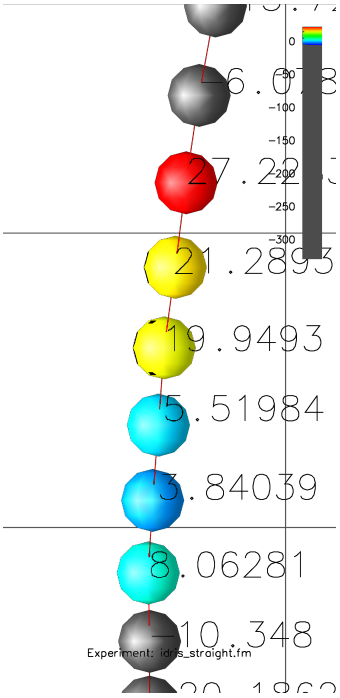
It is difficult to determine the exact reason why this performed better than the Linear model, and the difference is mostly quite small. What this does demonstrate however is that there are examples to be found in the results where the construction of a higher order, albeit 2^{nd} degree, model can closer match what the underlying image manifold is really doing. The other noteworthy point about this example is that there is not one, but two maxima in the values. This would most likely be the case if the underlying manifold is in some way curving, or the NURBS was overshooting.



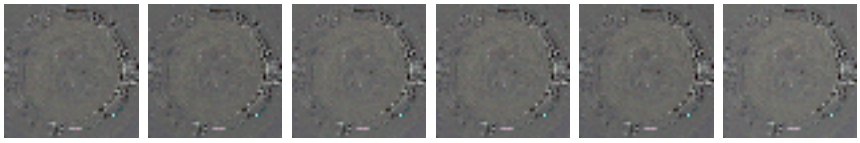
(a) Linear



(b) NURBS



(c) Detail section, positive means NURBS had less error



(d) Difference images for the detail stretch

Figure 11.1: Comparing Linear and NURBS (IDRIS_STRAIGHT, Euclidean, $g = 20$)

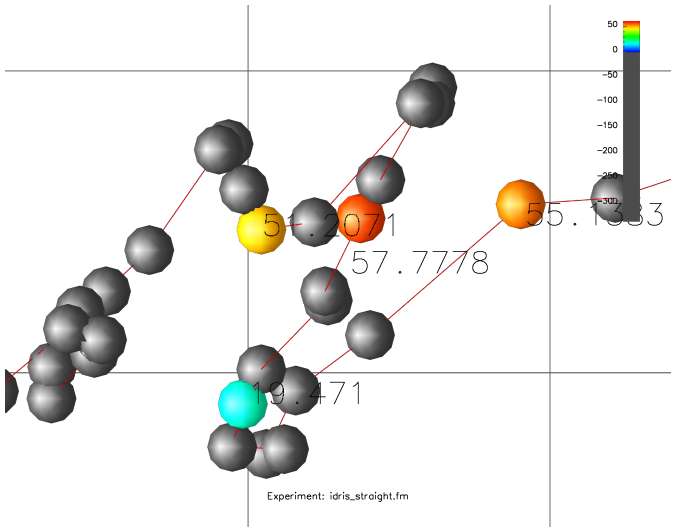


Figure 11.2: Comparing Linear and NURBS (IDRIS_STRAIGHT, Euclidean, $g = 2$)

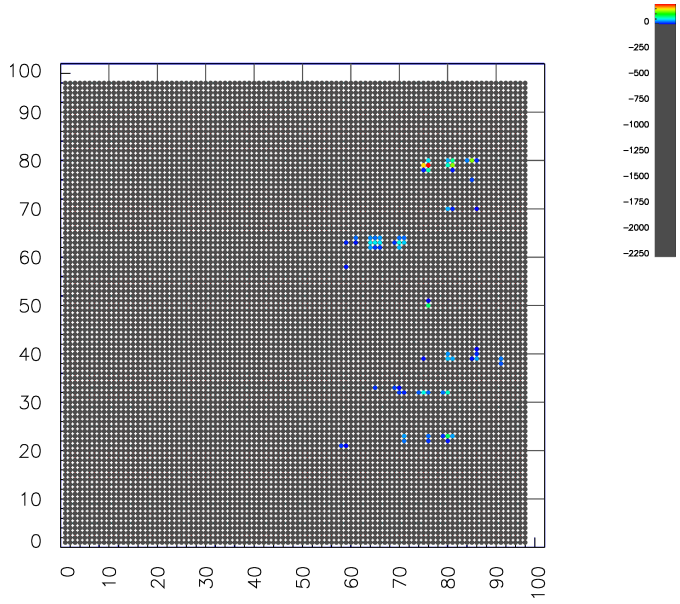
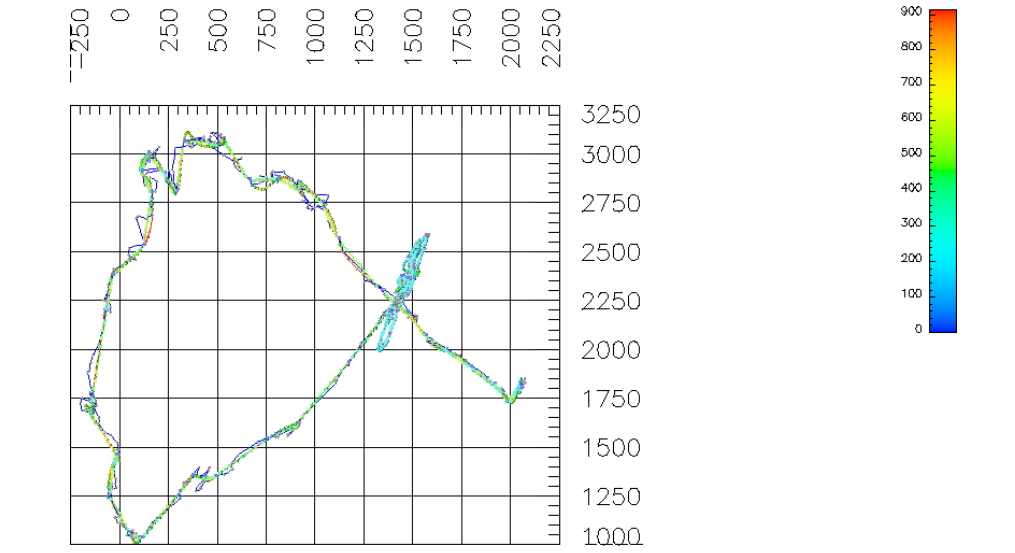


Figure 11.3: Comparing Linear and NURBS (2DWOODBOX, Euclidean, $g = 5$)

Examples of this can be found, even when $g = 2$. This is of particular interest, because it shows a case where introducing the continuity into the model has caused the model to better fit the data, even over short distances (we discussed previously how the manifolds were locally Euclidean). One example is shown in Figure 11.2. Further examples of this can be found, for instance Figure 11.3 shows the 2DWOODBOX manifold with the colour map only colouring instances of the NURBS model having the lowest error.

Further direct comparison between the Linear model and the NURBS model, as well as the original data, is possible, by showing all three projected into the same subspace. Figure 11.4 shows this for IDRIS_STRAIGHT, which was primarily selected because of the 'uncluttered' positioning of the data within the subspace[1]. Figure 11.4(a) shows an overview of all three, where it can be seen that in general they each follow an approximately similar path. Figure 11.4(b) shows a span where, by interpreting the colours of the glyphs, we can see that the NURBS model performs worse than the linear model. The shape of the span at this point in the NURBS case is influenced by its neighbours. Thus the requirement for the curve to be smooth has caused the path to deviate from the path taken by the linear model and this has not resulted in an improvement in terms of measured error. Figure 11.4(c) shows two further spans. In the span located in the top half of the image the paths taken by both the NURBS model and the linear model appear to be quite similar and the reported errors are correspondingly similar. Notice though that this is only three of the principle components of the data and what is happening in the other dimensions not represented in this visualisation could be different. In the span found in the lower half of the image the continuity requirements of the NURBS curve again force it to take a path which moves it further away from the sampled images. In both these spans however it seems that there is potential for derivative information, if acquired accurately, to improve the construction of the NURBS model by further constraining the smooth path which is taken between sample points.

[1]Similar figures could be produced for 2-D an beyond, however these are hard to interpret on screen, let alone in printed form.



(a) Overview

(b) Detailed span

(c) Detailed span(2)

Figure 11.4: Comparing the linear model, NURBS model and the original data in the same visualisation (IDRIS_STRAIGHT, Euclidean, $g = 10$). Image from the linear model are represented by cubes, the NURBS model is represented by spheres and the original images by grey diamonds.

Table 11.1: Results from Linear compared to NURBS, taxi

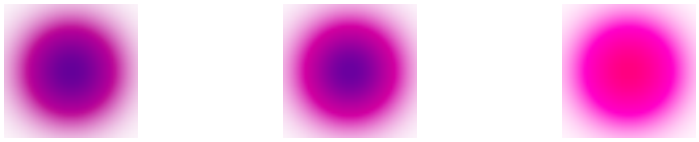
results of Linear compared to NURBS (taxi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2dteapot	−4786.70	−6471.00	−6908.80	−4466.70	−2744.90	−31309.30	missa
2dwoodbox	−13044.90	−17430.80	−16242.10	−14482.00	−8532.00	−289565.60	missa
Bmnoise	−17173.10	−12369.00	−756632.00	missa	missa	missa	missa
Brush	−4591.05	−3911.80	−4017.90	−4427.20	−4662.30	−2245.20	−237340.30
Chessboard	−36543.00	26957.00	11331.00	−5.00	−4218.00	missa	missa
Circle_3	−9275.10	−11931.60	−13696.20	−14121.00	−16184.90	−12719.00	−11189.00
Circle_4	−8713.30	−12059.90	−13653.20	−14874.60	−16885.40	−12715.00	−10632.00
Circle_5	−9031.00	−12356.70	−14419.50	−14250.20	−15625.80	−12767.00	−10244.00
Expt03	−5214.70	−127247.50	missa	missa	missa	missa	missa
Faces	−25618.00	−211664.00	missa	missa	missa	missa	missa
Idris_circle	−4061.40	−6617.60	−8873.00	−10681.10	−11987.60	−13411.80	−14158.00
Idris_fig8	−4698.00	−7112.80	−9462.70	−13053.50	−13128.10	−15411.20	−14771.00
Idris_straight	−3768.36	−3085.90	−2918.30	−3545.50	−3887.40	−5650.10	−4918.90
Knight_fighting	−1238.19	−1930.40	missa	missa	missa	missa	missa
Knight_kneeling	−1135.90	−1636.50	missa	missa	missa	missa	missa
Knight_standing	−1327.80	−580.80	missa	missa	missa	missa	missa
Sinecos	−18.06	−31.13	−12.00	21.25	−159.98	missa	missa
Straight_1	−4666.30	−4951.70	−5726.20	−6131.30	−6932.80	−6682.40	−7533.20
Straight_2	−4271.10	−4633.80	−5326.00	−5565.50	−5574.10	−5891.00	−5647.00
Straight_3	−4071.10	−4187.70	−4457.00	−4384.70	−4393.70	−3422.70	−3026.40
Straight_4	−4060.00	−4069.10	−4456.10	−4693.00	−4488.70	−3805.90	−2929.30
Straight_5	−4092.30	−4119.70	−4585.20	−5112.20	−4814.40	−4011.80	−3585.20
Straight_6	−4170.90	−4153.00	−4801.30	−4765.70	−5079.20	−3958.90	−3387.30
Straight_7	−4089.20	−4040.00	−4647.60	−5252.70	−4656.30	−4268.30	−3381.20
Straight_8	−4075.40	−4083.20	−4696.60	−5165.00	−5284.50	−5515.90	−4525.60
Woodbox	−6707.90	−7419.40	−8268.20	−9126.40	−9331.70	−10400.90	−11885.00

Table 11.2: Results from Linear compared to NURBS, pdiff

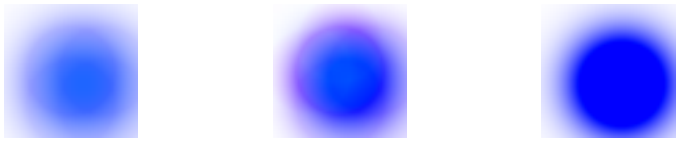
results of Linear compared to NURBS (pdiff)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2dteapot	−20.33	−20.18	−24.36	−25.53	−26.10	7.55	missa
2dwoodbox	−119.64	−141.47	−133.47	−111.04	−94.44	−306.53	missa
Bmnoise	−53.84	−9.10	−403.12	missa	missa	missa	missa
Brush	−54.22	−53.35	−46.34	−26.58	−31.56	−48.00	−1292.24
Chessboard	−150.87	191.96	81.54	0.00	170.15	missa	missa
Circle_3	−115.67	−143.56	−140.33	−105.77	−98.07	−75.34	−57.31
Circle_4	−116.77	−144.65	−142.12	−105.98	−100.04	−79.11	−58.09
Circle_5	−115.31	−141.89	−135.09	−107.26	−97.84	−82.86	−63.52
Expt03	−35.17	−112.96	missa	missa	missa	missa	missa
Faces	−90.65	−554.42	missa	missa	missa	missa	missa
Idris_circle	−121.55	−148.14	−152.82	−151.68	−153.89	−144.75	−119.54
Idris_fig8	−127.87	−156.61	−164.47	−171.89	−167.39	−161.87	−137.36
Idris_straight	−107.70	−124.88	−122.73	−117.19	−117.11	−129.04	−95.43
Knight_fighting	−3.17	−11.90	missa	missa	missa	missa	missa
Knight_kneeling	−3.64	−26.09	missa	missa	missa	missa	missa
Knight_standing	−6.87	−21.10	missa	missa	missa	missa	missa
Sinecos	−0.05	0.00	−0.03	−0.05	−0.24	missa	missa
Straight_1	−111.86	−125.57	−129.65	−104.74	−95.56	−65.80	−40.10
Straight_2	−109.48	−128.16	−125.66	−110.86	−96.39	−64.82	−43.94
Straight_3	−119.25	−134.23	−139.03	−119.51	−101.61	−81.51	−50.29
Straight_4	−120.05	−134.15	−136.53	−111.99	−106.11	−91.98	−44.91
Straight_5	−119.60	−131.46	−135.47	−115.67	−103.28	−77.39	−38.48
Straight_6	−114.92	−127.59	−133.01	−111.27	−103.70	−75.30	−48.56
Straight_7	−117.40	−129.16	−138.84	−116.87	−104.18	−88.54	−42.89
Straight_8	−120.94	−134.06	−143.29	−128.20	−119.68	−110.12	−43.43
Woodbox	−45.93	−64.98	−68.71	−70.40	−69.22	−69.83	−72.01

Table 11.3: Results from the 4-D Gauss dataset

g	Metric	Linear	NURBS	Difference
$g = 5$	Euclidean	$\mu = 1802.66$, $\sigma = 774.07$	$\mu = 1475.89$, $\sigma = 677.485$	326.77
$g = 5$	Taxi	$\mu = 103046$, $\sigma = 47594.5$	$\mu = 79244$, $\sigma = 40742.7$	23802
$g = 5$	pdiff	$\mu = 1002.71$, $\sigma = 489.382$	$\mu = 996.153$, $\sigma = 527.299$	6.557



(a) Linear, 3616.92 (b) NURBS, 3142.21 (c) Original



(d) Linear, 5068.07 (e) NURBS, 3428.91 (f) Original

Figure 11.5: Linear, NURBS and original for 4-D dataset (Gauss). Euclidean distance from the original image is reported.

11.1.1 4-D dataset

We introduced in Chapter 3 a 4-D dataset which was created by translating the centre of a Gaussian across the image plane and altering its colour. The results from this dataset are show in Table 11.3.

There are no parameter space visualisations for this dataset as this would require performing some dimensionality reduction on the parameter space itself. There is also no corresponding PCA projections from image space either because of the problems introduced by the size of this dataset and the fact that even in these PCA plots it is not possible to extract useful information.

Figure 11.5 shows the output from the Linear model, from NURBS and the original image for two points on the Gauss manifold. In both cases NURBS reported a lower error than the Linear model, which agrees with a perceptual assessment. In the case of the first example, Figure 11.5(b), NURBS has better captured the 'peak' in intensity and the distribution as a whole[2]. In the case of

[2]This is visually quite subtle and subject to printing issues however it is clear on a well calibrated display and readers of the printed version are encouraged to obtain an electronic

the second example, Figure 11.5(e), it seems that NURBS has not only better represented the change in intensities, but the change in position too. Some ghosting is still visible, however the shape is clearer than in the linear image. This trend is seen across the results from this dataset.

11.2 Linear vs. PDE

Direct comparisons between the PDE model and the other models are slightly complicated by the fact that the PDE model outputs fewer images for a given input than the other models. This is because the first and last few inputs (the exact number depends upon the order of the PDE used) are required to construct derivative estimates for building the model. In the cases where the 2^{nd} order model performed best this is not a problem.

As was the case when we compared the Linear results with the NURBS results we see that the Linear model also overwhelmingly outperforms the PDE model. Likewise though it is possible to find counter examples to this trend that show that even the simplistic approach to modelling image manifolds we have employed, can, potentially show improvements.

Here we offer a number of examples to support this assertion. Figure 11.6 shows these examples, which are all taken from the configuration $N = 6$, $\alpha = 1$, $o = 2$ and the remainder term enabled, because this is the best performing of the $o = 2$ configurations, which are also directly compatible with the Linear model. Positive values in Figure 11.6 indicate the areas where the PDE model outperformed the Linear one. The maximum difference in favour of the PDE model shown here is of the order of 500. With the Euclidean model the *mean* error reported was 1327.5, which seems to suggest that this is potentially significant.

Given that in Chapter 10 we saw the numeric solution of the PDE represented a poorer, but nonetheless still viable alternative to the analytic solution, the results

copy.

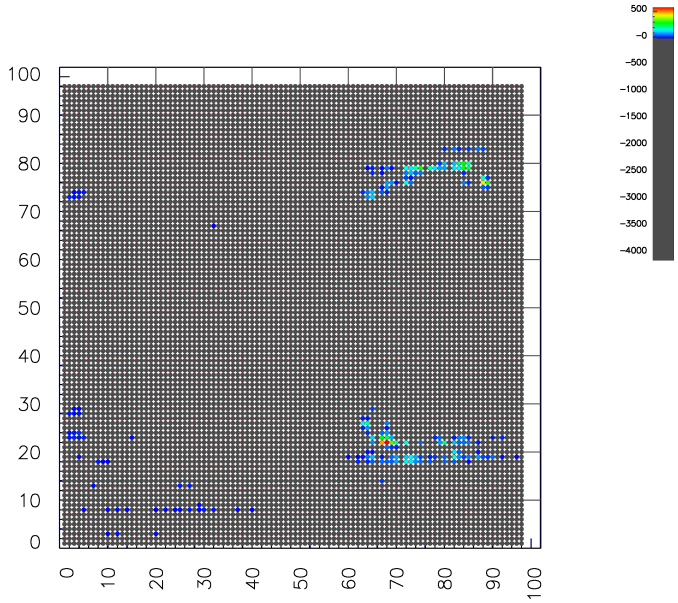


Figure 11.6: Comparing Linear and PDE (2DWOODBOX, Euclidean, $g = 5$)

from it have been omitted here in favour of the analytic solution. The tables are still presented in Appendix H, but it is no great surprise to see that the results of the comparison between linear and the numeric solution are higher than for the analytic solution.

Table 11.4: Results from Linear compared to PDE analytic, taxi

results of Linear compared to PDE analytic (taxi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DTEAPOT	−10995.80	−8535.70	−6431.10	−6724.50	miss2	−7400.30	missa
2DWOODBOX	−27584.90	−23914.00	−24032.10	−23071.00	miss2	−47492.60	missa
BMNOISE	−56647.10	−60605.00	−49463.00	missa	missa	missa	missa
KNIGHT_FIGHTING	−6602.29	−4306.10	missa	missa	missa	missa	missa
KNIGHT_KNEELING	−8484.60	−5173.90	missa	missa	missa	missa	missa
KNIGHT_STANDING	−15631.00	−5716.20	missa	missa	missa	missa	missa
SINECOS	−81.91	−101.72	−36.64	−163.09	miss2	missa	missa

Table 11.5: Results from Linear compared to PDE analytic, pdiff

results of Linear compared to PDE analytic (pdiff)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DTEAPOT	−57.33	−36.47	−21.82	−16.21	miss2	−25.78	missa
2DWOODBOX	−40.78	−37.23	−80.91	−120.36	miss2	−316.38	missa
BMNOISE	−1139.71	−546.80	−438.66	missa	missa	missa	missa
KNIGHT_FIGHTING	−19.12	−11.08	missa	missa	missa	missa	missa
KNIGHT_KNEELING	−21.32	−19.86	missa	missa	missa	missa	missa
KNIGHT_STANDING	−39.00	−23.61	missa	missa	missa	missa	missa
SINECOS	−0.09	0.08	0.17	−0.25	miss2	missa	missa

11.3 NURBS vs. PDE

Tables 11.6 and 11.7 show a comparison between the NURBS and PDE models. In these tables a positive value represents the PDE model outperforming the NURBS model. Results for other metrics are shown in Appendix H.

For the comparison between NURBS and PDE we again face the problem that the higher order PDE models do not produce the same number of outputs.

In general these comparisons seem to indicate that the PDE model has outperformed the NURBS model where the gaps are smaller (e.g. $g = 2$, $g = 5$). With the larger gaps there are two things to consider. Firstly our results from the PDE model were somewhat incomplete due to excessive run times. This incompleteness can be seen in Table G.3, where Bmnoise only has 4 completed experiments for $g = 10$ for instance. Secondly, as the gaps get larger the results become less meaningful anyway, and eventually are completely dominated by the aliasing.

One possible explanation as to why the PDE model has outperformed the NURBS model generally could be that the use of the derivative information in the PDE model has helped to constrain the surfaces built to pass through more plausible regions of image space. From our experiments it is not possible to confirm or discard this theory, for that reason we have included the use of derivative information to improve NURBS models in Chapter 13 as future work.

Table 11.6: Results from PDE analytic compared to NURBS, taxi

results of PDE analytic compared to NURBS (taxi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2dteapot	6209.10	2064.70	−477.70	2257.80	miss1	−23909.00	missa
2dwoodbox	14540.00	6483.20	7790.00	8589.00	miss1	−242073.00	missa
Bmnoise	39474.00	48236.00	−707169.00	missa	missa	missa	missa
Knight_fighting	5364.10	2375.70	missa	missa	missa	missa	missa
Knight_kneeling	7348.70	3537.40	missa	missa	missa	missa	missa
Knight_standing	14303.20	5135.40	missa	missa	missa	missa	missa
Sinecos	63.85	70.58	24.64	184.34	miss1	missa	missa

Table 11.7: Results from PDE analytic compared to NURBS, pdiff

results of PDE analytic compared to NURBS (pdiff)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2dteapot	37.00	16.30	−2.53	−9.32	miss1	33.34	missa
2dwoodbox	−78.87	−104.24	−52.56	9.32	miss1	9.85	missa
Bmnoise	1085.87	537.70	35.54	missa	missa	missa	missa
Knight_fighting	15.95	−0.82	missa	missa	missa	missa	missa
Knight_kneeling	17.67	−6.23	missa	missa	missa	missa	missa
Knight_standing	32.13	2.50	missa	missa	missa	missa	missa
Sinecos	0.04	−0.08	−0.19	0.20	miss1	missa	missa

11.4 Conclusion

Both PDE an NURBS are viable candidates for numerical models of image manifolds. These are both computationally viable, if a little slower, although the numeric PDE solver is markedly slower than the analytic one. Speed issues are not primarily a concern here — the research is more open and what is considered prohibitively slow today can be optimised and will naturally run significantly faster on the hardware of tomorrow anyway.

Both of these solutions provide the ability to model higher order curves and surfaces. Without the presence of additional information (e.g. calculated derivatives, not estimated from the boundaries) or a new approach to building the curves and surfaces, they fall short of offering any substantial improvements over our simple linear model.

This chapter may appear quite negative — other than the 4-D Gauss dataset we do not have any instances where the mean error from the linear model is greater than the best mean error from either of our other models. The fact that we can find examples where the higher order models do offer a measurable improvement, in identical or comparable setups is however very encouraging, especially when viewed alongside the results from the 4-D dataset. This suggests that with an improved understanding of image manifolds, and application of that knowledge to the model construction it may yet be possible to build models that better capture what is happening in image space than the simple linear model.

Another possible approach would be a hybrid model — using the output from the 'best' model over different stretches of the manifold. This could be pre-measured, or alternatively estimated, depending upon the application.

Results here do not represent the best achievable results from the NURBS or PDE models. We therefore devote the Chapter 12 to developing and testing two ideas for improving these models.

11.5 Summary

In this chapter we have seen:

- comparisons of the results with different models from a number of datasets;
- how the Linear model usually outperforms simple application of the NURBS and PDE models;
- a number of instances where this is not the case;
- a discussion of these results, and the implications for our work.

In the next chapter we will explore:

- another way of building NURBS curves;
- a curvature directed method for constructing models;
- some example results from this.

Chapter 12

Improving the models

We have seen previously how our higher order interpolation of the image manifolds fails to produce the improvement we had hoped it would. This we have suggested is caused at least in part by the effects of the noise within the images we have used. It is possible that even the quantisation of the images influences this somewhat! Furthermore fitting high order models to what is inherently sparsely sampled data is always going to be fraught with the danger of forcing the curve/surface to take an obscure route in order to fit the requirements we have imposed upon it.

Here we introduce two possible improvements to our NURBS model. The first improvement we offer comes from the fact that previously, in Chapter 9, Section 9.2.2.3, we touched briefly on the idea of *approximation* instead of interpolation. This would confer us a number of potential advantages, which could potentially help to overcome some of the problems which could have prevented our higher order models from showing an improvement. First, and most notably this would offer some basic form of insulation from the noise in our datasets. Secondly, if applied with care this could also help avoid the overshooting we have seen in our evaluation of the NURBS interpolation scheme we used.

The second improvement we introduce comes from Chapter 6, where we introduced a method for measuring the non-straightness of our manifolds: midpoint distance. We propose to use the results of this midpoint distance measure to di-

rect the sampling strategy employed to create the surface model. This potentially could allow us to reduce the average error in a surface by concentrating sampling at the areas where it is most needed. We test this method here with both an approximation, as well as an interpolation based surface construction.

Why did we not repeat these improvements for the PDE model too? Firstly, we must provide explicit boundary conditions to both PDE surface variants we used; these have to be regularly spaced for both the numeric and analytic solutions. In effect we are forced into regular sampling for the PDE model[1]. Secondly though we already truncate the Fourier series at some point, and we already stop the iterative procedure at a fixed value, so in that respect what we do with the PDE model could be argued to be more of an approximation than an interpolation anyway!

This chapter is intended to provide 'proof of concept' for both the approximation approach and the use of measurements from the manifolds to guide construction. Since the aim of this chapter is to show how a simple measurement can be used to improve the models we build we do not consider the curvature measure introduced. This is because, although it could be relevant, incorporating tangent information into NURBS curve construction could be performed better than by estimating curvature and is beyond the scope of the current research. It is however, an interesting avenue for further research. Furthermore, as we noted in our discussion of midpoint distance and curvature measures the curvature is likely to understate large differences (Figure 6.11(c)) because of the normalisation term, which are clearly important for defining which points to retain to build a model.

12.1 Approximation

As with Chapter 9 we use (Piegl and Tiller, 1997) as our primary reference. In this the authors divide the approximation algorithms into two categories: algorithms

[1]There are a number of possible ways to address this, discussed later on.

which start with very few control points and repeatedly add more control points, until the measured error is within acceptable bounds; or algorithms which start with many control points and discard as many control points as possible. These are referred to by Piegl and Tiller (1997) as type 1 and type 2 algorithms respectively.

In the case of the type 1 algorithms, the minimum number of control points we can use is $p + 1$, where p is the degree of the curve. A curve can be fitted using linear least squares fitting at this point, and the error measured for each span. The individual treatment of each span is important here — clearly there is no point in improving the curve in a span with an acceptable error just because there remains at least one more span for which the error is too large. Knots are added at the centre of every span for which the error is too large and the process repeated.

For the type 2 case least squares fitting is still employed, however the reverse procedure is employed; a curve is built and then knot removal algorithms are used to remove knots in appropriate places, where the error will remain tolerable even after the removal. The algorithm we use is a 3$^{\text{rd}}$-party implementation of (Piegl and Tiller, 1997, A9.10), which is part of the 'NURBS++'[2] library. Here the curve starts out as a 1$^{\text{st}}$ degree curve, the knots suitable for removal are removed, and then the degree of the curve is raised if it is less than the target degree. This procedure is repeated until the degree has been raised to the desired degree.

12.1.1 Experimental setup

In testing this we build an individual curve for each pixel of the datasets in question. This has several possible advantages. Firstly, we can re-use the existing implementations, which make a number of assumptions about the space being at most three dimensional. For an RGB image each pixel is a point in a three dimensional space. Secondly, this allows us to test an alternative strategy we had

[2] http://sourceforge.net/projects/libnurbs/

not previously considered. Potentially some pixels in some of our datasets will require significantly less knots than others, e.g. the background of an image does not change much between images. Of course by doing this we are assuming that each pixel is independent, something which clearly is not always the case.

The main questions we wish to answer here are firstly, 'does this offer an improvement over global interpolation?' Secondly, 'how does this new scheme perform with higher degree curves?' Additionally, 'how many control points actually get used to represent these approximated curves, once the algorithm has finished?'

The experiments we run here are much more limited in scope than the experiments we used for Chapter 9. Here we wish to show the feasibility of constructing per pixel approximations of the manifold. To this end we only consider one dataset, namely BRUSH, and we only run experiments at the $g = 5$ scale. We picked the BRUSH dataset, because it is one of the 1D datasets (required for the third party library), and one which we considered explicitly when we discussed midpoint distance earlier, in Chapter 6. The gap $g = 5$ was chosen because in the linear results (Chapter 8) it did not lose the shape of the manifold in the PCA plots, yet it has a relatively high percentage of new views synthesised in the output.

12.1.2 Results and Discussion

The first experiment we ran varied the maximum permissible error for the approximation. The degree of the curve was fixed at $p = 2$. Figure 12.1 shows the error in the produced images as reported by the Euclidean distance metric on the left axis, and the average number of control points used per dimension on the right axis. For reference the equivalent experiment with the Linear model produced a mean error $\mu = 335.5$, and for the global NURBS interpolation the best was $\mu = 479.712$, which was a 2^{nd} degree curve.

What Figure 12.1 shows is not really particularly surprising. The trend in

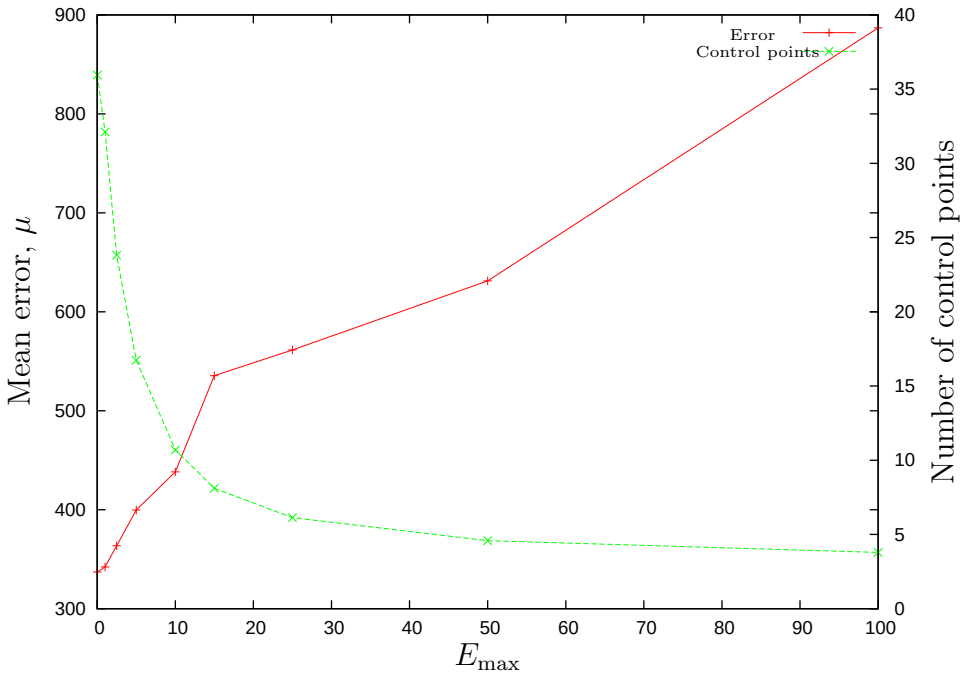


Figure 12.1: Maximum error in approximation, $p = 2$, $g = 5$

the number of control points in the generated surface appears to tail off towards some limit. This limit will be the minimum number of control points possible for a 2nd degree surface, i.e. 3, although with a maximum allowable error from the approximation of 100 it does not reach this limit. This seems to be visible in the graph, and by the time $E_{\max} = 255$ we would expect it to have hit this limit. For the most part, the maximum allowable error from the approximation is proportional to the measured error in the generated images, with a notable deviation around the $E_{\max} = 15$ point. It would seem therefore that at around this point there is a threshold or cutoff, which with the $p = 2$ curve causes a number of particularly important control points to be able to be discarded by the algorithm (this is not the only instance where we see such behaviour, see Figure 12.3). $E_{\max} = 0$ is a special case which is equivalent to interpolation, although not directly the same as the previous scheme we used since there may be more intermediate control points generated by the approximation algorithm.

The best error from the $p = 2$ case is only fractionally higher than for the Linear model, and much lower than the best of the global interpolation NURBS results.

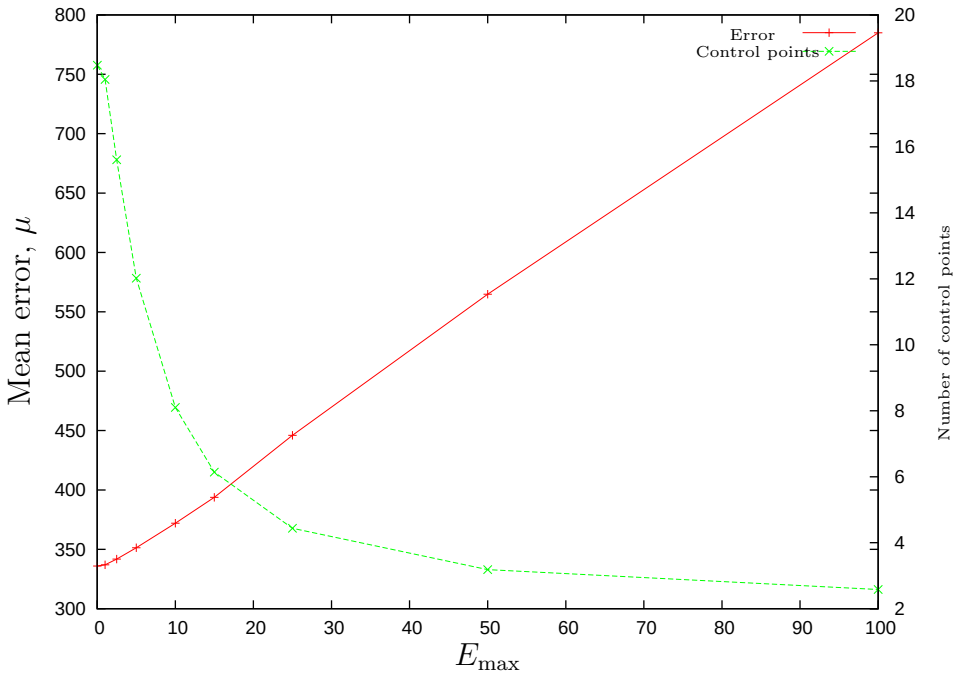


Figure 12.2: Maximum error in approximation, $p = 1$, $g = 5$

This would appear to corroborate what we stated in Chapter 9 earlier about the global interpolation scheme likely introducing undue 'wiggles' and contortions into the curves and surfaces generated.

When we reduce the degree of the curve to $p = 1$, i.e. linear, as shown in Figure 12.2 we can observe several things. Firstly, the measured results are only fractionally worse than the Linear model ($\mu = 335.9$ compared to $\mu = 335.5$). Secondly, exactly as would be expected, the measured error in the output images is proportional to the maximum allowed error in the curve generation.

When we increase the degree however to $p = 3$ the results are somewhat less intuitive. Firstly, it should be noted that in a number of instances, around the $E_{\max} = 5$ the approximation completely failed, and an error was returned by the NURBS++ library. At a number of other nearby points, e.g. $E_{\max} = 10, 15$, the measured error is again particularly high. This also coincides with the area during which the most control points are being discarded by the implementation. It also coincides with the anomalous area we observed in Figure 12.1. This seems to suggest that there are still sufficiently many control points as to force the curve to

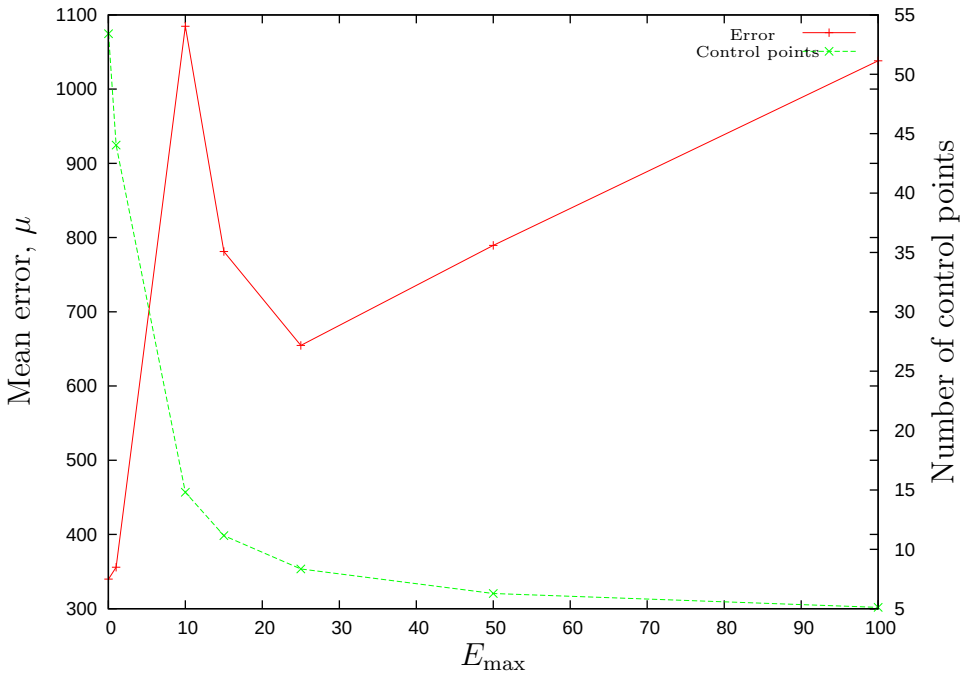
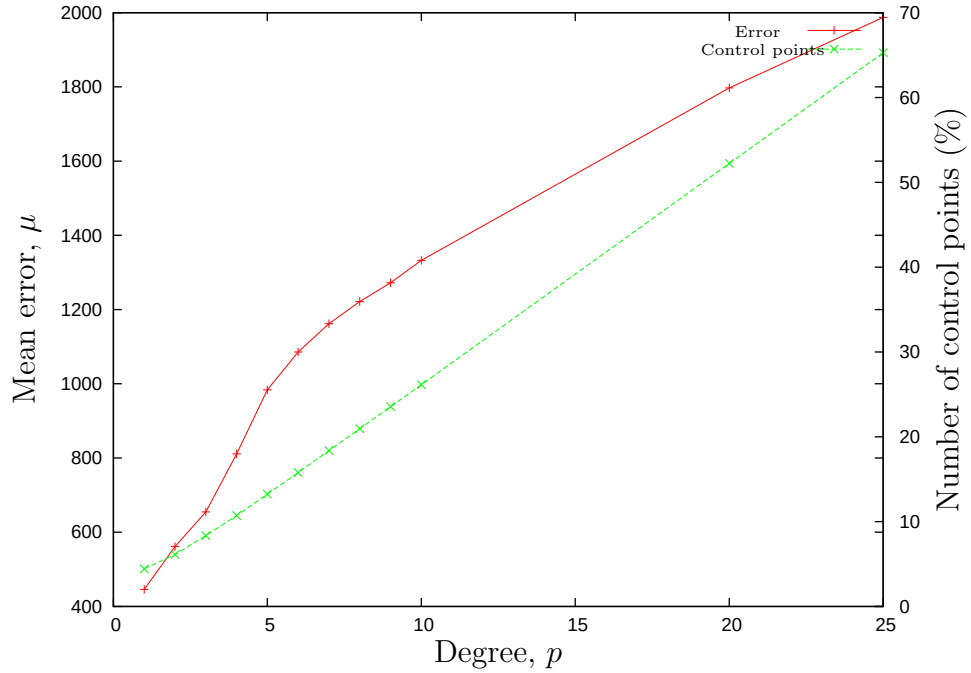


Figure 12.3: Maximum error in approximation, $p = 3$, $g = 5$

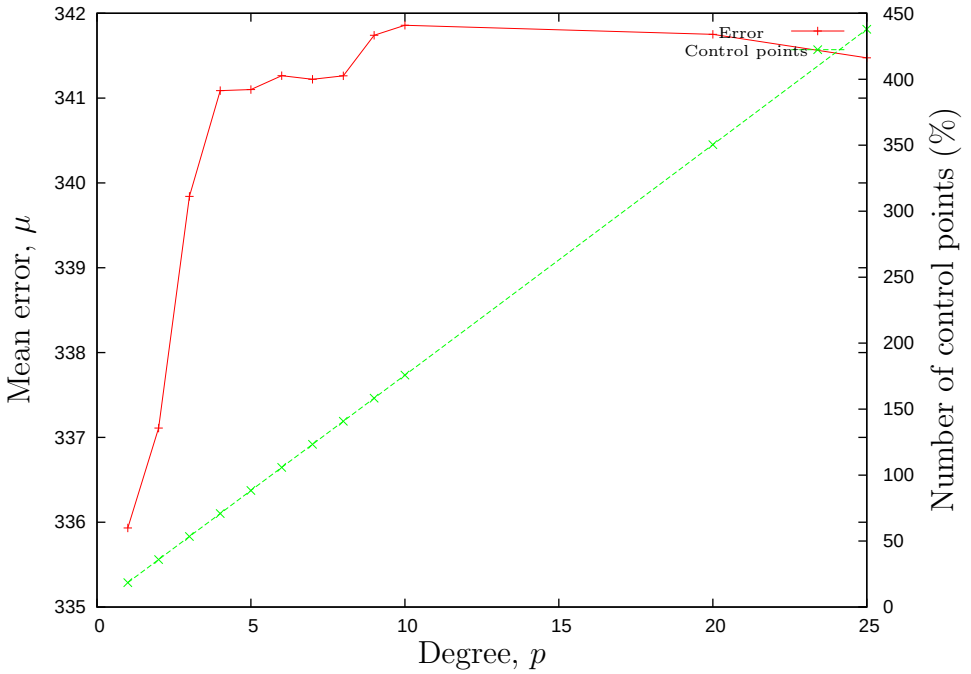
take a contorted path, to meet the continuity requirements of a 3rd degree curve, and the positional requirements. At a lower $E_{\max}$ many more control points are used, and the distance each single point is expected to represent in image space is far smaller, allowing for much less freedom in the overall shape of the curve.

The final issue we wish to consider with this new, per-pixel approximation scheme, is the degree of the curve produced by the approximation. In Figure 12.4 we consider this for two values of $E_{\max}$. The results for Figure 12.4(a) show the same general trend as we observed for the global interpolation with NURBS and the PDE method (e.g. Figures 10.19 and 10.20, page 225), that is to say the error seems to be a roughly linear function of the degree of the curve.

Interestingly though the same cannot be said for Figure 12.4(b), where the error actually peaks, and then appears to dip again towards the end. The most likely explanation for this seems to come from the number of control points that get used to build the curve. By the time we reach $p = 4$, where the error starts to flatten out we already have as many control points as we do by the $p = 25$ in Figure 12.4(a). This means that the effect of any one control point will be



(a) $E_{\max} = 25$



(b) $E_{\max} = 0.01$

Figure 12.4: Curve degree in approximation, $g = 5$

felt over a much smaller section of the total curve, and it also means that any 'wiggles' to meet the continuity requirements can occur over very short stretches of the curve. Furthermore the small value of $E_{\max}$ means that our curve must pass very close to the known points. This combined with the small area of effect and the fact that a straight line would be an easy way to satisfy continuity requirements suggests that the curves after each known sample bend very sharply, before taking a straighter path for the remainder of the gap to the next sample. Contrast this with the global interpolation scheme we previously used with NURBS, where the dominating factor in the path taken seemed to be meeting the continuity requirements imposed.

12.2 Midpoint distance directed construction

The other angle we wish to investigate is using the midpoint distance measure we introduced in Chapter 6 to control our sampling strategy instead of relying upon even spacing. To do this is actually quite simple in practice: we always include a sample point at the start ($u = 0$), and discard samples until the measured midpoint distance exceeds some predetermined threshold.

This has another advantage, besides the obvious one of directing our surface construction towards less straight parts. This will almost certainly not result in a uniform $\bar{u}$, the curve parametrisation, which means we can also avoid the uniform knot vector by using averaging.

The basic algorithm we use is shown in Algorithm 12.1, where n is the total number of samples available. markkeep and markskip are functions which mark a given sample for retention or discarding respectively. Our midpoint distance estimation procedure relies on the existence of at least one additional sample within the range for which we wish to estimate the midpoint distance. This slightly complicates things because we must now only start measuring the midpoint distance after already having passed over one sample. We decide if we should use

the sample in question or not based upon what happens to the sample immediately following it.

Algorithm 12.1 Midpoint distance directed sampling

```
l ← 1 {Index of last kept point}
i ← 3 {Current position, ignore the first two points}
while i ≤ n do
  while i ≤ n − 1 AND md(l, i) < max do
    if i − 2 = l then
      markskip i − 1
    end if
    markskip i
    i ← i + 1
  end while
  markkeep i
  l ← i
  i ← i + 2
end while
markkeep n {Always keep the last point}
```

12.2.1 Experimental setup

The experiments for testing this idea are mostly designed to show feasibility of using midpoint distance to direct surface modelling. As with the approximations earlier, we restrict ourselves to only using the Brush dataset as an illustration. We used an adapted version of the NURBS++ library, identical to the one used for the approximations, except for the modifications required to accept a list of points to use or skip whilst building the curve. For the global interpolation we use a different implementation than the one we used in Chapter 9, which was easier to adapt to this scheme, by virtue of the fact that it only supports curves[3], and not the generalisations we introduced in Chapter 9.

We run both the interpolation and the approximations with maximum mid-

[3]By way of comparison here we also used this implementation, which is far simpler to understand, to verify some of our observations from the earlier NURBS chapter. Comparable results were obtained for the higher degree curves, with the same problems being evident, this despite using a different solver for the system of equations, as well as a different implementation of the basis functions.

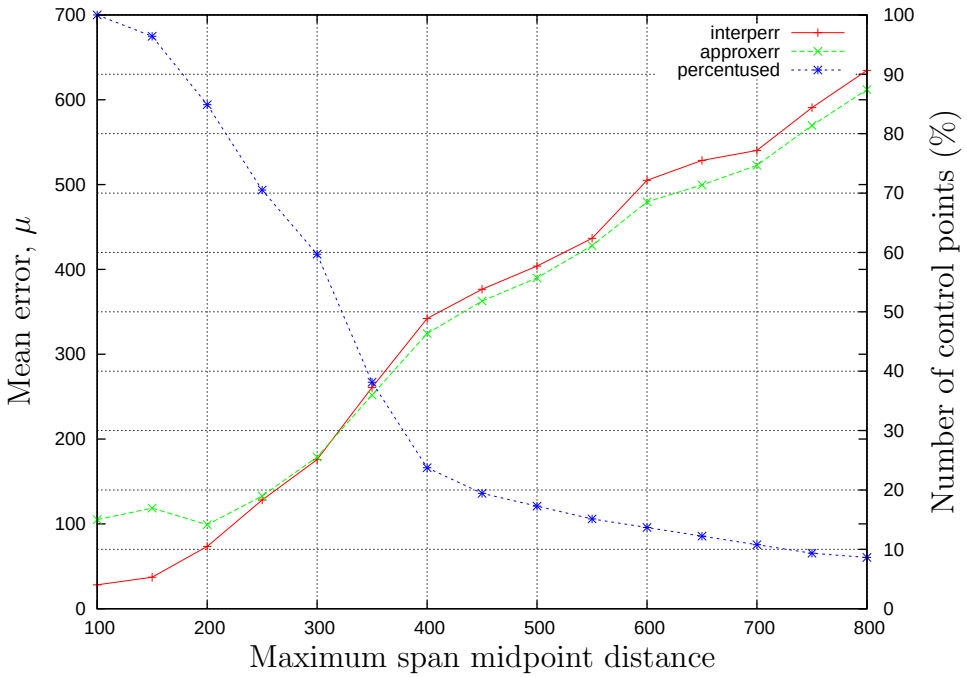


Figure 12.5: Midpoint distance based curve construction, $E_{\max} = 1$, Euclidean

point distances starting at 100 and incrementing by 50 all the way to 800. From these we counted the number of samples that were skipped, and recorded it as a percentage. The midpoint distance based sampling algorithm can be applied with either interpolation or approximation based NURBS, and therefore only one percentage needs to be recorded at each stage. The degree was fixed as $p = 2$ throughout. This avoids introducing any extra complications into these results, but still has some level of smoothness more than the Linear model. Additionally this matches the findings from Chapter 9 with regards to the degree, where $p = 2$ consistently outperformed almost all of the alternative configurations, with almost all of the metrics.

12.2.2 Results and Discussion

When maximum span midpoint distance is 700, the closest equivalent linear result, in terms of number of samples used ($g = 10$), is remarkably close. The result from the Linear model at $g = 10$ is $\mu = 528.7$, but the results from the midpoint distance directed NURBS approximation is $\mu = 522.6$, and the global NURBS

interpolation was $\mu = 680.9$. At 700 though there are still slightly more samples being used in the construction of the curve, so it is worth looking at 750 as well, which has $\mu = 569.7$, but uses slightly fewer samples.

When the maximum span midpoint distance is chosen to be 450 the number of samples that will be retained is very close to $g = 5$. At $g = 5$ with the BRUSH dataset the error in the Linear model is $\mu = 335.5$, with the midpoint distance directed NURBS approximation this error is $\mu = 362.8$, and for global NURBS interpolation an error of $\mu = 479.7$ is reported. This actually uses slightly less input samples than the $g = 5$ configuration.

At somewhere near when the maximum span midpoint distance is 300 the interpolation starts to have a higher error than the approximation does. This potentially tell us something about the noise inherent within the samples themselves, although this crossover point is inextricably related to many other factors making it hard if not impossible to draw any conclusions about the noise in the images themselves from this.

Interestingly, if we repeat this experiment with $p = 1$, i.e. the same as the Linear model, the results still do not outperform the results from the Linear model. There are a number of possible explanations for this. Firstly, because we consider every available sample this means that we model a slightly longer manifold than the Linear model does, in this instance it is two more samples. This was part of the motivation for comparing results based on percentages of samples used, and not absolute values. Secondly, by sampling non-uniformly we will see the error increase at some specific points of the manifold, where we are now sampling less than previously. If the increased error in these areas is larger than the reduction in error in the areas we now use more samples from then we will of course see a net increase in the average error.

12.3 Conclusions

We have seen here how the midpoint distance estimate we introduced in Chapter 6 can be used to improve the output of at least the NURBS model, and it seems likely it would also be applicable to the other models. Midpoint distance is just one example of tangible way for us to understand what is actually happening in image space. There are other properties of image manifolds (and manifolds in general) which could potentially be used to improve the sampling strategies, in a similar way to the way we have used the midpoint distance here. For example the curvature or the topological properties of the image manifold could help improve the sampling strategy; if we discovered that there was a hole in the manifold, for example, it might make sense to ensure that we use samples from around the edge of the hole when we build our models.

We have presented here some preliminary results from two possible methods for improving the results we saw in Chapter 9. These results are promising — from both of the ideas we tested we clearly saw improvements. Furthermore this makes a strong case for the possibility to improve, potentially significantly, upon the results we presented in Chapter 9, with other improvements to the method we use to build the models.

12.4 Summary

In this chapter we have seen:

- a proposed method using approximation to improve the NURBS results;
- how the estimations of midpoint distance we used can improve our results overall;
- some preliminary testing of this.

In the next chapter we will:

- look back at the research question;
- draw this thesis to a close with a number of general comments;
- suggest some areas of potential follow up work.

Chapter 13

Conclusion

13.1 Answering the question

In Section 4.2 we introduced our research question. Since then, throughout this thesis we have conducted experiments designed to enable us to answer that question. Our experiments have already implicitly answered this question, here we now explicitly answer it, and point to the relevent areas of the thesis.

The research question primarily asked "is it possible to use extended variants of generic geometric modelling techniques, to construct vectorial representations of the appearance of scenes?". The evidence we have presented, from the three vectorial models we have proposed points to a cautiously optimistic yes. We have successfully built and tested a number of vectorial models of image manifolds in image space. The results from these tests have been somewhat encouraging, the simple linear model of Chapter 8 indicated there were a number of instances where, with appropriate sampling, the visual quality may be sufficient for use in a number of application domains, even IBR. Whilst the NURBS and PDE models we introduced and tested in Chapters 9 and 10 respectively failed to show the major improvements we had hoped to see over the linear model they where an improvment in a small number of cases and they did show that modelling using such structures in image space is feasible. In some cases improvements were seen

(e.g. Section 11.1.1 as well as small individual cases). Perhaps more importantly though than the numeric results, is what we saw in Chapter 12, which suggested that there was potential to improve on these methods.

In terms of the aims which we outlined in Chapter 4 we have clearly met both of them. We have considered the properties (geometry and curvature in Chapter 6, as well as more general visualisations in Chapter 7) and have considered the theory of image manifolds in the early parts of the thesis, discussing image manifold related literature in Chapter 3; we have proposed a new approach to modelling image manifolds exactly using new extensions of highly generic methods in Part III of the thesis.

13.2 General Conclusion

Image manifolds are difficult objects to deal with; they are large, and like any real-world dataset, algorithms for working with them are fraught with difficulties. The potential benefits from modelling image manifolds are enormous however, and this thesis makes some progress towards realising that goal. The possible applications for image manifolds range from computer graphics problems like IBR, to appearance based vision problems and beyond. The concepts and algorithms involved could be generalised further and applied to more problem domains, beyond the field of computing entirely. Even within computing and the vision and graphics domains the promise of a cheap, photo realistic model of the appearance of a real object, or environment, complete with correct global illumination, is enormous.

The progress we have made towards the geometric modelling of image manifolds has been significant. We have demonstrated the computational feasibility of global, higher order models of structures in image space. Numerically the results from the more sophisticated models have been outperformed by the simple linear model, although in many circumstances the gap between them is small, and in some specific, small subsets of the data the higher order models have actually

outperformed the linear model.

This fact alone is quite promising, we have seen clear evidence that there are scenarios where we can improve on the linear model. Furthermore, we have seen several methods which improve upon our initial results. These methods suggest a number of possible ways of improving the results even further. Of particular interest in this area are the derivatives of the image manifolds, which would open up a number of avenues for improving our results.

Our approach to this problem has been rigorous throughout and our methodology has been firmly guided by the principles of scientific method. We have relied heavily on metrics through out this thesis, because of the scale and quantity of images involved. Yet even the most perceptual of these metrics is still only a relatively simplistic model of the human visual system itself. It is possible that a user study of an application of this work would reveal a different conclusion than the almost purely numeric conclusion we have arrived at, and we have seen some evidence of this in our discussion.

13.3 Future work

There are a number of areas which we believe would now merit further investigation. Firstly, our numeric solution to the PDEs of Chapter 10 does not reflect current state of the art PDE solvers. Whilst this is not a problem for our work it would nonetheless be interesting to investigate a number of other techniques for solving the PDEs, including finite element methods instead of finite difference methods and meshless methods.

One interesting idea that has arisen from this work is the idea of directly connecting the multigrid solution of the PDEs to the rendering process, using the intermediate, coarser grids as lower detail versions of the finished article. This would allow for sensible, dynamic level of detail mechanisms which are used in many rendering applications.

Throughout this thesis we have restricted ourselves to datasets which can be approximately thought of as being uniformly sampled. This made sense from a practical perspective, simplifying comparison and implementation of the methods we have considered. This need not however always be the case. Finite element methods would enable us to have non-uniform discrete domains and hence non-uniform boundary conditions, whilst a similar approach could be taken with the NURBS model.

There are a number of ways in which we could potentially improve our model construction, in particular the NURBS model; for instance if we understood better the higher order derivatives of our manifolds at sampled points we could constrain the construction of our models to better reflect what is really happening in image space, and not merely 'join the dots'. Good derivative information would also benefit other areas of our work, for example the curvature estimation process.

During the study of the midpoint distance measure we observed a number of interesting features. The triangles we measured are typically close to being equilateral (Figures 6.11(c) and 6.11(d) seem to be the common cases). A further general observation arises from the study of the midpoint distances (height), compared to the width of the triangles. This is illustrated in Figure 13.1, which shows a scatter plot of width and height of triangles from the BRUSH dataset with $g = 1$. This seems to point towards a highly complex "helix" structure. The evidence for this helix structure comes from the clear general trend for height to be approximately half of the width, which indicates that the path being taken completes a full rotation approximately every four images (owing to the approximately right angled corners in the triangles). The only way to avoid looping (which we clearly are not observing in the images) therefore is for this rotation to be part of a helix like structure. It seems reasonable though to conclude from this crude analysis that there are some deeper geometric implications for the data, which deserve further study. The observations seem to point to something which is too regular to be the

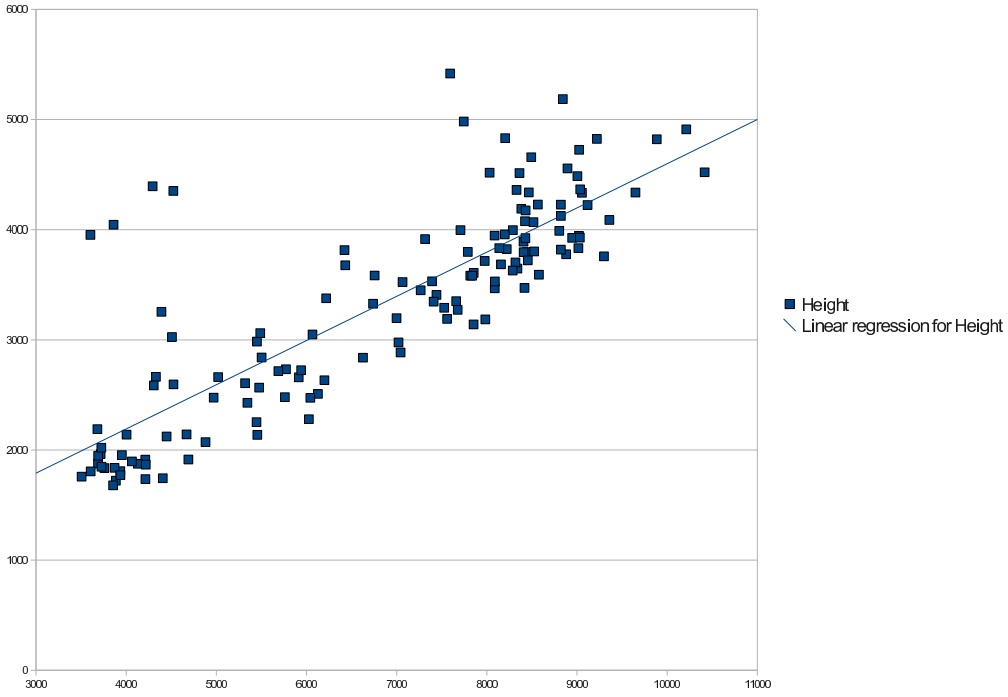


Figure 13.1: Scatter plot of width vs. height for BRUSH

result of noise. This observation of a helix like structure also agrees with what Lu (1998) reported. The fact that our PCA plots did not reveal this indicates that the helix is observed only comparatively locally and raises a number of questions. Firstly, how much of the helix is actually perceptually important compared to just the "spine" of the helix and how much of the helix shape is simply ignored by methods like GTM (Bishop et al., 1998*b*) and those proposed by Verbeek (2006)? Conversely, how much of this helix structure can be captured by exact representations of the manifold? How does this change as the scale (gap, g) is varied? Also, can knowledge of the helix structure itself be exploited to improve the modelling process further? Can we visualise this structure?

Our midpoint distance measurements were only constructed with the Euclidean distance metric in mind. There are a number of potential problems with computing the midpoints of other metrics, however we believe that this is an area suitable for further work.

The improvements we presented in Chapter 12 have only been tested at the level of proof of concept and a more thorough, and extended study of these should

be undertaken. Additionally the method we used to measure the deviation from straightness of a span used a maximum midpoint distance, however a cumulative measure may also prove to be useful.

The topology of the image manifolds is an interesting, and open area of study. In particular the connections between the (potentially complex) topology of the actual objects or environment in a scene and the images in image space is of interest. Further investigation of this for our datasets would likely prove to be illuminating, and could be used to further improve the sampling strategy, as we saw in Chapter 12.

To date we have also largely avoided placing our work in any specific application domain, but there are a number of graphics and vision problems we would like to use this work to address. A study in one or more of these areas would be most interesting. An application in the graphics domain would also raise some interesting HCI oriented problems, such as the seamless linking of totally disjointed, but physically connected image manifolds.

There is always scope for more, interesting and varied datasets to be considered, especially synthetic ones which exhibit certain features of interest. Of particular interest would be applications involving other forms of data, not just images. Some possible examples might include audio, or even gene sequences! We have only considered images in the RGB colour space, whereas data in other colour spaces, particularly those that are perceptually linear may prove to be interesting.

Additionally for each of the datasets we have studied we have only had one sample at each point on the manifold, yet we know that most of our datasets contain some noise. Given the potential impact this noise has upon attempts to model the data another interesting task to perform for the datasets would be to generate multiple samples at each location on the manifold at a much denser sampling. In practice a 1-D manifold with noise is likely to not be a single path through image space, but a probability density field, or 'sausage' shape in image

space. At the moment we have only one path, not necessarily coinciding with the spine of the sausage itself. Taking this knowledge into account may help to further improve results in the future.

Final thoughts

Throughout the course of this thesis we have made a number of novel contributions. Primarily these are:

- The PDE surface method in higher dimensions, and higher order variants is new to the graphics domain, with the one noted exception (Du and Qin, 2007) in the 3-D case.
- The extension and generalisation of NURBS to higher dimensional spaces, seems to be novel beyond the 6-D case.
- The proposal of modelling structures like image manifolds using extended versions of classic, generic, graphics techniques.
- We have obtained and presented a large number of varied and new datasets which we argue represent image manifolds.

The results and conclusions may not be a revolution and are not going to change any of the potential application domains overnight, but they are very much an evolution, and a step forward; they have opened up a number of avenues of further study which would not have been possible without such groundwork.

Bibliography

Adelson, E. H. and Bergen, J. R. (1991), *The Plenoptic Function and the Elements of Early Vision*, MIT Press, pp. 3–20.

Agarwala, A., Zheng, K. C., Pal, C., Agrawala, M., Cohen, M., Curless, B., Salesin, D. and Szeliski, R. (2005), Panoramic video textures, *in* 'SIGGRAPH '05: ACM SIGGRAPH 2005 Papers', ACM, New York, NY, USA, pp. 821–827.

Aharon, M. and Kimmel, R. (2006), "Representation analysis and synthesis of lip images using dimensionality reduction", *Int. J. Comput. Vision* , Vol. 67, Kluwer Academic Publishers, Hingham, MA, USA, pp. 297–312.

Aliaga, D. G. and Carlbom, I. (2001), Plenoptic stitching: a scalable method for reconstructing 3d interactive walk throughs, *in* 'SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques', ACM Press, New York, NY, USA, pp. 443–450.

Alotaibi, N. (2009), Image Region Completion by Structure Reconstruction and Texture Synthesis, PhD thesis, Computer Science department, Aberystwyth University, Aberystwyth, United Kingdom.

Arandjelovic, O., Shakhnarovich, G., Fisher, J., Cipolla, R. and Darrell, T. (2005), Face recognition with image sets using manifold density divergence, *in* 'CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1', IEEE Computer Society, Washington, DC, USA, pp. 581–588.

Bar-Joseph, Z., El-Yaniv, R., Lischinski, D. and Werman, M. (2001), "Texture mixing and texture movie synthesis using statistical learning", *IEEE Transactions on Visualization and Computer Graphics* , Vol. 7, IEEE Educational Activities Department, Piscataway, NJ, USA, pp. 120–135.

Barten, P. G. J. (1990), "Evaluation of subjective image quality with the square-root integral method", *J. Opt. Soc. Am. A* , Vol. 7, OSA, pp. 2024–2031.

Basharat, A. and Shah, M. (2009), Time series prediction by chaotic modeling of nonlinear dynamical systems, *in* 'ICCV', IEEE.

Belkin, M. and Niyogi, P. (2003), "Laplacian eigenmaps for dimensionality reduction and data representation", *Neural Comput.* , Vol. 15, MIT Press, Cambridge, MA, USA, pp. 1373–1396.

Bengio, Y., Paiement, J., Vincent, P., Delalleau, O., Le Roux, N. and Ouimet, M. (2004), Out-of-sample extensions for LLE, isomap, MDS, eigenmaps, and spectral clustering, *in* S. Thrun, L. Saul and B. Schölkopf, eds, 'Advances in Neural Information Processing Systems 16', MIT Press, Cambridge, MA.

Berard, A. D. (1971/72), "Characterizations of metric spaces by the use of their midsets: Intervals", *Fundamenta Mathematicae* , Vol. 73, pp. 1–7.

Bichsel, M. and Pentland, A. P. (1994), "Human face recognition and the face image set's topology", *CVGIP: Image Underst.* , Vol. 59, Academic Press, Inc., Orlando, FL, USA, pp. 254–261.

Bishop, C. M., Svensén, M. and Williams, C. K. I. (1996), Gtm: A principled alternative to the self-organizing map, *in* 'ICANN 96: Proceedings of the 1996 International Conference on Artificial Neural Networks', Springer-Verlag, London, UK, pp. 165–170.

Bishop, C. M., Svensén, M. and Williams, C. K. I. (1998*a*), "Developments of the generative topographic mapping", *Neurocomputing* , Vol. 21, pp. 203–224.

Bishop, C. M., Svensén, M. and Williams, C. K. I. (1998*b*), "Gtm: The generative topographic mapping", *Neural Computation* , Vol. 10, pp. 215–234.

Blau, B., Dodsworth, C., Branagan, L., Ippolito, J., Musgrave, K. and Waggenspack, W., eds (1996), *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA.

Bloor, M. I. G. and Wilson, M. J. (1990), "Using partial differential equations to generate free-form surfaces", *Comput. Aided Des.* , Vol. 22, Butterworth-Heinemann, Newton, MA, USA, pp. 202–212.

Boeing Information and Support Services (n.d.), 'DTNURBS'. Website offline (http://ocean.dt.navy.mil/dtnurbs/).

Bozkaya, T. and Ozsoyoglu, M. (1999), "Indexing large metric spaces for similarity search queries", *ACM Trans. Database Syst.* , Vol. 24, ACM Press, New York, NY, USA, pp. 361–404.

Bregler, C. and Omohundro, S. M. (1994), Nonlinear image interpolation using manifold learning, *in* G. Tesauro, D. S. Touretzky and T. K. Leen, eds, 'NIPS', MIT Press, pp. 973–980.

Briggs, W. L., Henson, V. E. and McCormick, S. F. (2000), *A Multigrid Tutorial*, Other Titles in Applied Mathematics, second edn, SIAM.

Cai, D., He, X. and Han, J. (2007), Regularized regression on image manifold for retrieval, *in* 'MIR '07: Proceedings of the international workshop on Workshop on multimedia information retrieval', ACM, New York, NY, USA, pp. 11–20.

Campbell, F. W. and Robson, J. G. (1968), "Application of fourier analysis to the visibility of gratings.", *J. Physiol.* , Vol. 197, pp. 551–566.

Carlsson, G., Ishkhanov, T., Silva, V. and Zomorodian, A. (2008), "On the local behavior of spaces of natural images", *Int. J. Comput. Vision* , Vol. 76, Kluwer Academic Publishers, Hingham, MA, USA, pp. 1–12.

Cevikalp, H., Verbeek, J., Jurie, F. and Kläser, A. (2008), Semi-supervised dimensionality reduction using pairwise equivalence constraints, *in* 'International Conference on Computer Vision Theory and Applications', pp. 489–496.

Chang, C.-Y., Maciejewski, A. A., Balakrishnan, V., Roberts, R. G. and Saitwal, K. (2007), "Quadtree-based eigendecomposition for pose estimation in the presence of occlusion and background clutter", *Pattern Anal. Appl* , Vol. 10, pp. 15–31.

Cheatham, R. M., Red, W. E. and Jensen, C. G. (2005), "Direct process control using n-dimensional nurbs curves", *Computer-Aided Design and Applications* , Vol. 2, pp. 825–834.

Chen, H. and Varshney, P. K. (2003), "Mutual information based image registration for remote sensing data.", *International Journal of Remote Sensing* , Vol. 24, pp. 3701–3706.

Chen, S. E. and Williams, L. (1993), View interpolation for image synthesis, *in* 'SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques', ACM Press, New York, NY, USA, pp. 279–288.

Cho, E., Kim, D. and Lee, S.-Y. (2003), Posed face image synthesis using nonlinear manifold learning, *in* J. Kittler and M. S. Nixon, eds, 'AVBPA', Vol. 2688 of *Lecture Notes in Computer Science*, Springer, pp. 946–954.

Churchill, R. V. and Brown, J. W. (1978), *Fourier Series and Boundary Value Problems*, McGraw-Hill, Inc.

Copson, E. (1968), *Metric spaces*, number 57 *in* 'Cambridge tracts in mathematics and mathematical physics', Camgidge University Press.

Costantini, R., Sbaiz, L. and Süsstrunk, S. (2008), "Higher order svd analysis for dynamic texture synthesis", *IEEE Transactions on Image Processing* , Vol. 17, pp. 42–52.

Daly, S. (1993), The visible differences predictor: an algorithm for the assessment of image fidelity, *in* 'Digital images and human vision', MIT Press, Cambridge, MA, USA, pp. 179–206.

de Ridder, D., Kouropteva, O., Okun, O., Pietikinen, M. and Duin, R. P. (2003), "Supervised locally linear embedding", Vol. 2714, pp. 333—341.

Dixon, M., Jacobs, N. and Pless, R. (2006), Finding minimal parameterizations of cylindrical image manifolds, *in* 'CVPRW '06: Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop', IEEE Computer Society, Washington, DC, USA, p. 192.

Dollár, P., Rabaud, V. and Belongie, S. (2006), Learning to traverse image manifolds, *in* 'NIPS'.

Donoho, D. L. and Grimes, C. (2003), "Hessian eigenmaps: New locally linear embedding techniques for high-dimensional data", *Proceedings of the National Academy of Sciences of the United States of America* , Vol. 100.

Donoho, D. L. and Grimes, C. (2005), "Image manifolds which are isometric to Euclidean space", *Journal of Mathematical Imaging and Vision* , Vol. 23, Kluwer Academic Publishers, Norwell, MA, USA, pp. 5–24.

Doretto, G., Chiuso, A., Wu, Y. N. and Soatto, S. (2003), "Dynamic textures", *Int. J. of Computer Vision* , Vol. 51, pp. 91–109.

Du, H. and Qin, H. (2007), "Free-form geometric modeling by integrating parametric and implicit PDEs", *IEEE Transactions on Visualization and Computer Graphics* , Vol. 13, IEEE Computer Society, Los Alamitos, CA, USA, pp. 549–561.

Efros, A. A. and Leung, T. K. (1999), "Texture synthesis by non-parametric sampling", *ICCV* , Vol. 02, IEEE Computer Society, Los Alamitos, CA, USA, p. 1033.

Elgammal, A. (2004), Nonlinear generative models for dynamic shape and dynamic appearance, *in* 'CVPRW '04: Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04) Volume 12', IEEE Computer Society, Washington, DC, USA, p. 182.

Elgammal, A. and Lee, C. (2008), The role of manifold learning in human motion analysis, *in* 'Human Motion Understanding, Modelling, Capture, and Animation', Springer, p. 2.

Elgammal, A. and Lee, C.-S. (2004), "Inferring 3d body pose from silhouettes using activity manifold learning", *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on* , Vol. 2, IEEE Computer Society, Los Alamitos, CA, USA, pp. 681–688.

Elgammal, A. and Lee, C.-S. (2007), "Nonlinear manifold learning for dynamic shape and dynamic appearance", *Comput. Vis. Image Underst.* , Vol. 106, Elsevier Science Inc., New York, NY, USA, pp. 31–46.

Elgammal, A. and Lee, C.-S. (2009), "Tracking people on a torus", *IEEE Trans. Pattern Anal. Mach. Intell.* , Vol. 31, IEEE Computer Society, Washington, DC, USA, pp. 520–538.

Fitzgibbon, A. W., Wexler, Y. and Zisserman, A. (2005), "Image-based rendering

using image-based priors", *International Journal of Computer Vision* , Vol. 63, pp. 141–151.

Girod, B. (1993), What's wrong with mean-squared error?, *in* 'Digital images and human vision', MIT Press, Cambridge, MA, USA, pp. 207–220.

Golub, G. and Van Loan, C. (1989), *Matrix Computations*, second edn, Johns Hopkins Press, Baltimore, MD.

Gortler, S. J., Grzeszczuk, R., Szeliski, R. and Cohen, M. F. (1996), The lumigraph, *in* (Blau et al., 1996), pp. 43–54.

Goshen, L., Shimshoni, I., Anandan, P. and Keren, D. (2005), "Motion recovery by integrating over the joint image manifold", *Int. J. Comput. Vision* , Vol. 65, Kluwer Academic Publishers, Hingham, MA, USA, pp. 131–145.

Hall, P. M., Marshall, A. D. and Martin, R. R. (1998), Incremental eigenanalysis for classification, *in* J. N. Carter and M. S. Nixon, eds, 'BMVC', British Machine Vision Association.

Ham, J., Ahn, I. and Lee, D. (2006), Learning a manifold-constrained map between image sets: applications to matching and pose estimation, *in* 'CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition', IEEE Computer Society, Washington, DC, USA, pp. 817–824.

He, X., Ma, W.-Y. and Zhang, H.-J. (2004), Learning an image manifold for retrieval, *in* 'MULTIMEDIA '04: Proceedings of the 12th annual ACM international conference on Multimedia', ACM Press, New York, NY, USA, pp. 17–23.

Hu, C., Chang, Y., Feris, R. and Turk, M. (2004), Manifold based analysis of facial expression, *in* 'CVPRW '04: Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04) Volume 5', IEEE Computer Society, Washington, DC, USA, p. 81.

Huang, L. and Su, C. (2006), "Facial expression synthesis using manifold learning and belief propagation", *Soft Comput.* , Vol. 10, Springer-Verlag, Berlin, Heidelberg, pp. 1193–1200.

Isaksen, A., McMillan, L. and Gortler, S. J. (2000), Dynamically reparameterized light fields, *in* 'SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques', ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, pp. 297–306.

Jenkins, O. C. (2003), Relative localization from pairwise distance relationships using kernel pca, Technical Report CRES-03-010, Center for Robotics and Embedded Systems, University of Southern California.

Jones, T., Carpenter, A. and Golland, P. (2005), Voronoi-based segmentation of cells on image manifolds, *in* 'Proc. ws. on Computer Vision for Biomedical Image Applications', pp. 535–543.

Kendall, M. G. and Stuard, A. (1968), *The advanced theory of statistics*, Vol. 3, second edn, Butler & Tanner Ltd.

Kim, S.-W., Aggarwal, C. C. and Yu, P. S. (2001), Effective nearest neighbor indexing with the Euclidean metric, *in* 'CIKM '01: Proceedings of the tenth international conference on Information and knowledge management', ACM Press, New York, NY, USA, pp. 9–16.

Kubiesa, S., Ugail, H. and Wilson, M. J. (2004), "Interactive design using higher order PDEs", *The Visual Computer* , Vol. 20, pp. 682–693.

Labrosse, F. (2003), On the editing of images: selecting, cutting and filling-in, *in* 'Proceedings of the International Conference on Vision, Video, and Graphics', Bath, UK, pp. 71–78.

Labrosse, F. (2006), "The visual compass: Performance and limitations of an appearance-based method", *J. Field Robotics* , Vol. 23, pp. 913–941.

Labrosse, F. (2007), "Short and long-range visual navigation using warped panoramic images", *Robotics and Autonomous Systems* .

Lee, J. A. and Verleysen, M. (2007), *Nonlinear Dimensionality Reduction*, Springer Publishing Company, Incorporated.

Levenshtein, V. I. (1966), Binary codes capable of correcting deletions, insertions, and reversals, Technical Report 8.

Levoy, M. and Hanrahan, P. (1996), Light field rendering, *in* (Blau et al., 1996), pp. 31–42.

Li, J., Hao, P. and Zhang, C. (2008), Transferring colours to grayscale images by locally linear embedding, *in* 'Proc. of the British Machine Vision Conference 2008'.

Li, M., Chen, X., Li, X., Ma, B. and Vitányi, P. (2003), The similarity metric, *in* 'SODA '03: Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms', Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, pp. 863–872.

Liu, C.-B., Lin, R.-S., Ahuja, N. and Yang, M.-H. (2006), Dynamic textures synthesis as nonlinear manifold learning and traversing, *in* 'The 17th British Machine Vision Conference', BMVA, pp. 859–868.

Lorensen, W. E. and Cline, H. E. (1987), "Marching cubes: A high resolution 3d surface construction algorithm", *SIGGRAPH Comput. Graph.* , Vol. 21, ACM, New York, NY, USA, pp. 163–169.

Lowe, D. G. (1999), Object recognition from local scale-invariant features, *in* 'Proc. of the Seventh IEEE International Conference on Computer Vision', Vol. 2, pp. 1150–1157 vol.2.

Lu, H.-m. (1998), Geometric Theory of Images, PhD thesis, University of California, San Diego.

Lu, H.-m., Fainman, Y. and Hecht-Nielsen, R. (1998), Image manifolds, *in* N. M. Nasrabadi and A. K. Katsaggelos, eds, 'Applications of Artificial Neural Networks in Image Processing III', Vol. 3307, SPIE, pp. 52–63.

Maes, F., Collignon, A., Vandermeulen, D., Marchal, G. and Suetens, P. (1997), "Multimodality image registration by maximization of mutual information.", *IEEE Trans. Med. Imaging* , Vol. 16, pp. 187–198.

Martin, S. and Bäcker, A. (2005), Estimating manifold dimension by inversion error, *in* 'SAC '05: Proceedings of the 2005 ACM symposium on Applied computing', ACM, New York, NY, USA, pp. 22–26.

Martin, W. and Cohen, E. (2001), Representation and extraction of volumetric attributes using trivariate splines: a mathematical framework, *in* 'SMA '01: Proceedings of the sixth ACM symposium on Solid modeling and applications', ACM, New York, NY, USA, pp. 234–240.

Meytlis, M. and Sirovich, L. (2007), "On the dimensionality of face space", *IEEE Transactions on Pattern Analysis and Machine Intelligence* , Vol. 29, IEEE Computer Society, Los Alamitos, CA, USA, pp. 1262–1267.

Mitchel, T. and Labrosse, F. (2004), Visual homing: a purely appearance-based approach, *in* 'Proc. of Towards Autonomous Robotic Systems'.

Moghaddam, B. (1999), Principal manifolds and bayesian subspaces for visual recognition, *in* 'ICCV '99: Proceedings of the International Conference on Computer Vision-Volume 2', IEEE Computer Society, Washington, DC, USA, p. 1131.

Murase, H. and Nayar, S. K. (1995), "Visual learning and recognition of 3-D objects from appearance", *International Journal of Computer Vision* , Vol. 14, Kluwer Academic Publishers, Hingham, MA, USA, pp. 5–24.

Nayar, S., Nene, S. and Murase, H. (1996), "Subspace methods for robot vision", *IEEE Transactions on Robotics and Automation* , Vol. 12, pp. 750–758.

Neal, M. and Labrosse, F. (2004), Rotation-invariant appearance based maps for robot navigation using an artificial immune network algorithm., *in* 'Proceedings of the Congress on Evolutionary Computation', Portland, Oregon, USA.

Neumann, L., Matkovic, K. and Purgathofer, W. (1997), Perception based color image difference, Technical Report TR-186-2-97-21, Institute of Computer Graphics and Algorithms, Vienna University of Technology, Favoritenstrasse 9-11/186, A-1040 Vienna, Austria.

Nilsson, J., Sha, F. and Jordan, M. I. (2007), Regression on manifolds using kernel dimension reduction, *in* 'ICML '07: Proceedings of the 24th international conference on Machine learning', ACM, New York, NY, USA, pp. 697–704.

Pappas, T. and Safranek, R. (2000), Perceptual criteria for image quality evaluation, *in* 'Handbook of Image and Video Processing', 1^{st} edn, Academic press, chapter VIII, pp. 669–687.

Peyré, G. (2007), Texture synthesis and modification with a patch-valued wavelet transform, *in* 'Proc. of the Scale Space Variational Methods Conf.', pp. 640–651.

Peyré, G. (2008), "Image Processing with Non-local Spectral Bases", *SIAM Journal on Multiscale Modeling and Simulation* , Vol. 7, pp. 703–730.

Phillips, P. M. and Watson, G. (2003), Generalising video textures, *in* 'TPCG '03: Proceedings of the Theory and Practice of Computer Graphics 2003', IEEE Computer Society, Washington, DC, USA, p. 8.

Piegl, L. and Tiller, W. (1997), *The NURBS book*, 2nd edn, Springer.

Poland, J. and Zeugmann, T. (2006), Clustering the google distance iwth eigenvecotrs and semidefinite programming, Technical Report N-14, Division of Computer Science, Hokkaido Univserity.

Portilla, J. and Simoncelli, E. P. (2000), "A parametric texture model based on joint statistics of complex wavelet coefficients", *Int. J. Comput. Vision* , Vol. 40, Kluwer Academic Publishers, Hingham, MA, USA, pp. 49–70.

Press, W. H., Vetterling, W. T., Teukolsky, S. A. and Flannery, B. P. (2002), *Numerical Recipes in C++: the art of scientific computing*.

Rademacher, P. and Bishop, G. (1998), Multiple-center-of-projection images, *in* 'SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques', ACM Press, New York, NY, USA, pp. 199–206.

Ramasubramanian, M., Pattanaik, S. N. and Greenberg, D. P. (1999), A perceptually based physical error metric for realistic image synthesis, *in* 'SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques', ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, pp. 73–82.

Reisel, R. B. (1982), *Elementary theory of metric spaces: a course in constructing mathematical proofs*, Springer-Verlag.

Robert, P. and Richard, S. (2009), "A survey of manifold learning for images", *IPSJ Transactions on Computer Vision and Applications* , Vol. 1, pp. 83–94.

Rovatti, R., Borgatti, M. and Guerrieri, R. (1998), "A geometric approach to maximum-speed n-dimensional continuous linear interpolation in rectangular grids", *IEEE Trans. Comput.* , Vol. 47, IEEE Computer Society, Washington, DC, USA, pp. 894–899.

Roweis, S. T. and Saul, L. K. (2000), "Nonlinear dimensionality reduction by locally linear embedding", *Science* , Vol. 290, pp. 2323–2326.

Russakoff, D. B., Tomasi, C., Rohlfing, T. and Jr., C. R. M. (2004), Image similarity using mutual information of regions., *in* 'ECCV (3)', pp. 596–607.

Schödl, A., Szeliski, R., Salesin, D. and Essa, I. A. (2000), Video textures, *in* 'SIGGRAPH', pp. 489–498.

Shade, J. (2002), Approximating the plenoptic function, Technical report, University of Washington.

Shade, J., Gortler, S. J., wei He, L. and Szeliski, R. (1998), Layered depth images., *in* 'SIGGRAPH', pp. 231–242.

Shon, A. P., Grochow, K., Hertzmann, A. and Rao, R. P. N. (2005), Learning shared latent structure for image synthesis and robotic imitation., *in* 'NIPS'.

Shum, H. and Kang, S. B. (2000), Review of image-based rendering techniques., *in* K. N. Ngan, T. Sikora and M.-T. Sun, eds, 'VCIP', Vol. 4067 of *Proceedings of SPIE*, SPIE, pp. 2–13.

Shum, H.-Y. and He, L.-W. (1999), Rendering with concentric mosaics, *in* 'SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques', ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, pp. 299–306.

Shum, H.-Y., Wang, L., Chai, J.-X. and Tong, X. (2002), "Rendering by manifold hopping", *Int. J. Comput. Vision* , Vol. 50, Kluwer Academic Publishers, Hingham, MA, USA, pp. 185–201.

Souvenir, R. and Pless, R. (2007), "Image distance functions for manifold learning", *Image Vision Comput.* , Vol. 25, Butterworth-Heinemann, Newton, MA, USA, pp. 365–373.

Souvenir, R., Zhang, Q. and Pless, R. (2006), Image manifold interpolation using free-form deformations, *in* 'ICIP', IEEE, pp. 1437–1440.

Spivak, M. (1979), *A Comprehensive Introduction to Differential Geometry*, Vol. I, 2^{nd} edn, Publish or Perish, Inc.

Sturzl, W. and Mallot, H. (2006), "Efficient visual homing based on Fourier transformed panoramic images", *Robotics and Autonomous Systems* , Vol. 54, pp. 300–313.

Su, C. and Huang, L. (2005), Facial expression hallucination, *in* 'WACV-MOTION '05: Proceedings of the Seventh IEEE Workshops on Application of Computer Vision (WACV/MOTION'05) - Volume 1', IEEE Computer Society, Washington, DC, USA, pp. 93–98.

Szummer, M. and Picard, R. W. (1996), Temporal texture modeling, *in* 'ICIP', pp. 823–826.

Tenenbaum, J. B. (1998), Mapping a manifold of perceptual observations, *in* 'NIPS '97: Proceedings of the 1997 conference on Advances in neural information processing systems 10', MIT Press, Cambridge, MA, USA, pp. 682–688.

Tenenbaum, J. B., de Silva, V. and Langford, J. C. (2000), "A global geometric framework for nonlinear dimensionality reduction", *Science* , Vol. 290, pp. 2319–2323.

Thomas, J. (1995), *Numerical Partial Differential Equations: Finite Difference Methods*, Vol. 22 of *Texts in Applied Mathematics*, second edn, Springer.

Tian, C., Fan, G. and Gao, X. (2008), Multi-view face recognition by nonlinear tensor decomposition, *in* 'ICPR', IEEE, pp. 1–4.

Tino, P., Nabney, I., Sun, Y. and Williams, B. (2002), A principled approach to interactive hierarchial non-linear visualization of high-dimensional data, *in*

E. Wegman, A. Braverman, A. Goodman and P. Smyth, eds, 'Proceedings of the 33rd Symposium on the Interface', Vol. 33 of *Frontiers in Data Mining and Bioinformatics*, pp. 580–587.

Turk, M. A. and Pentland, A. P. (1991*a*), Face recognition using eigenfaces, *in* 'Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition', pp. 586–591.

Turk, M. and Pentland, A. (1991*b*), "Eigenfaces for recognition", *Journal of Cognitive Neuro Science* , Vol. 3, pp. 71–86.

Tustison, N. J. and Amini, A. A. (2004), Myocardial kinematics based on tagged mri from volumetric nurbs models, *in* A. A. Amini and A. Manduca, eds, 'Medical Imaging 2004: Physiology, Function, and Structure from Medical Images', Vol. 5369, SPIE, pp. 22–33.

Ugail, H., Bloor, M. I. G. and Wilson, M. J. (1999), "Techniques for interactive design using the PDE method", *ACM Trans. Graph.* , Vol. 18, ACM Press, New York, NY, USA, pp. 195–212.

van der Maaten, L. J. P., Postma, E. O. and van den Herik, H. J. (2009), Dimensionality reduction: A comparative review, Technical report, Tilburg University.

Varga, R. S. (1962), *Matrix Iterative Analysis*, Prentice Hall, Englewood Cliffs, NJ, USA.

Verbeek, J. (2006), "Learning nonlinear image manifolds by global alignment of local linear models", *IEEE Trans. Pattern Anal. Mach. Intell.* , Vol. 28, IEEE Computer Society, Washington, DC, USA, pp. 1236–1250.

Verbeek, J., Roweis, S. and Vlassis, N. (2004), Non-linear cca and pca by alignment of local models, *in* 'Neural Information Processing Systems (NIPS)', pp. 297–304.

Victor, B. (1985), *Metric spaces: iteration and application*, Cambridge University Press.

Wakin, M., Donoho, D., Choi, H. and Baraniuk, R. (2005), The multiscale structure of non-differentiable image manifolds, *in* 'Optics & Photonics', San Diego, CA.

Wang, C., Zhao, J., He, X., Chen, C. and Bu, J. (2009), "Image retrieval using nonlinear manifold embedding", *Neurocomput.* , Vol. 72, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, pp. 3922–3929.

Wang, H. and Ahuja, N. (2005), Rank-r approximation of tensors: Using image-as-matrix representation, *in* 'CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2', IEEE Computer Society, Washington, DC, USA, pp. 346–353.

Wang, J. and Jiang, T. (2007), "Nonrigid registration of brain mri using nurbs", *Pattern Recogn. Lett.* , Vol. 28, Elsevier Science Inc., New York, NY, USA, pp. 214–223.

Wang, Y. Z. and Zhu, S. C. (2002), A generative method for textured motion: Analysis and synthesis, *in* 'ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part I', Springer-Verlag, London, UK, pp. 583–598.

Wang, Y. and Zhu, S.-C. (2004), "Analysis and synthesis of textured motion: Particles and waves", *IEEE Trans. Pattern Anal. Mach. Intell.* , Vol. 26, IEEE Computer Society, Washington, DC, USA, pp. 1348–1363.

Wei, L.-Y. and Levoy, M. (2000), Fast texture synthesis using tree-structured vector quantization, *in* 'SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques', ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, pp. 479–488.

Woodland, A., Ugail, H. and Labrosse, F. (2007), The PDE surface method in higher dimensions, *in* 'VVG'07 Workshop'.

Xu, G. and Bajaj, C. L. (2003), "Curvature computations of 2-manifolds in r^k", *J. Comp. Map* , Vol. 21, pp. 681–688.

Yee, Y. H. and Newman, A. (2004), A perceptual metric for production testing, *in* 'SIGGRAPH '04: ACM SIGGRAPH 2004 Sketches', ACM, New York, NY, USA, p. 121.

Zhang, Q. and Pless, R. (2005), Segmenting cardiopulmonary images using manifold learning with level sets, *in* Y. Liu, T. Jiang and C. Zhang, eds, 'CVBIA', Vol. 3765 of *Lecture Notes in Computer Science*, Springer, pp. 479–488.

Zhang, Q., Souvenir, R. and Pless, R. (2006), "On manifold structure of cardiac mri data: Application to segmentation", *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on* , Vol. 1, IEEE Computer Society, Los Alamitos, CA, USA, pp. 1092–1098.

Zhang, Q. and Xu, G. (2007), "Curvature computations for n-manifolds in rn+m and solution to an open problem proposed by r. goldman", *Comput. Aided Geom. Des.* , Vol. 24, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, pp. 117–123.

Zhou, G., Dong, N. and Wang, Y. (2009), "Non-linear dynamic texture analysis and synthesis using constrained gaussian process latent variable model", *Circuits, Communications and Systems, Pacific-Asia Conference on* , Vol. 0, IEEE Computer Society, Los Alamitos, CA, USA, pp. 27–30.

Zhou, H., Chen, M. and Webster, M. F. (2002), Comparative evaluation of visualization and experimental results using image comparison metrics, *in* 'VIS '02: Proceedings of the conference on Visualization '02', IEEE Computer Society, Washington, DC, USA.

Zomorodian, A. J., Ablowitz, M. J., Davis, S. H., Hinch, E. J., Iserles, A., Ockendon, J. and Olver, P. J. (2005), *Topology for Computing (Cambridge Monographs on Applied and Computational Mathematics)*, Cambridge University Press, New York, NY, USA.

Appendix A

Literature summary

This appendix serves as a summary of the notes that were made whilst reviewing the literature.

Some of these papers could be in several categories. Placement is effectively arbitrary, although based on personal view of best fit. Several substantially similar papers are grouped together. A number of arguments run through a large percentage of the papers discussed. These are covered in more depth in the main text. This is just a summary, and some parts are explained more fully in the original text.

A.1 Classic IBR

- MCOP (Rademacher and Bishop, 1998)

  Each column of an image has a different centre of projection.

  Specially modified camera required for acquisition, limited (intrinsic) dimensionality, per pixel depth.

- Layered Depth Images (Shade et al., 1998)

  Sparse sampling of light arriving at a point in space.

  Limited (intrinsic) dimensionality, only positions, depth information required.

- Plenoptic stitching (Aliaga and Carlbom, 2001)

  Select and merge pixels from omnidirectional images which form a closed loop in world space.

  Loop must be closed in world space, Camera must be omnidirectional and unwrapping known. Limited (intrinsic) dimensionality.

- Image-based priors (Fitzgibbon et al., 2005)

  Prior knowledge of the scene is used to constrain image generation.

  Probabilistic formulation of image synthesis.

- View Interpolation (Chen and Williams, 1993)

  Linear interpolation is used to update pixel positions and colours relative to the camera between frames.

  Accurately calibrated camera positions required, as is per pixel depth information. Limited intrinsic dimensionality, linear interpolation.

- Lightfield/Lumigraph (Gortler et al., 1996; Levoy and Hanrahan, 1996)

  Sample all rays of light arriving at (or leaving) a given point in space. New images are synthesised by combining appropriate subsets of rays.

  Assumes no occlusions. Limited to views looking at a single object, or views of a space from a fixed point.

- Concentric Mosaics (Shum and He, 1999)

  Rays of scene are sampled on various concentric circles, which are then combined to synthesise new views.

  Only horizontal parallax is captured. Space is assumed to be free of occlusions. Limited (intrinsic) dimensionality.

A.2 New IBR

- Temporal Texture modelling (Szummer and Picard, 1996)

  Textures are modelled as a linear spatio-temporal autoregressive model.

  This method assumes that the input is a multivariate Guassian distribution, with constant mean and covariance. Our proposed models do not make this assumption and are suited to modelling inputs which vary more than just temporally.

- Video textures (Phillips and Watson, 2003; Schödl et al., 2000)

  Video (or audio) frames are selected from a series based on Euclidean distance between all the candidate frames and the previous frame.

  This does not build a faithful model of a time series, instead it produces a plausible, continuous stream of frames selected from the input. Our models aim to faithfully model input, which can be more generic than just a time series.

- Panoramic video textures (Agarwala et al., 2005)

  Video stitching and looping to create the illusion of a continuous, spatially and temporally coherent panorama.

  This is solving a different problem than our models. Firstly the approach taken is that of stitching patches of images together in a seamless way. The resulting illusion of temporal coherence is only an illusion. In any synthesised frame the actual pixels shown may come from inputs acquired at very different times. This is only synthesising plausible panoramas, not faithfully modelling an input image manifold. The camera is fixed in one position and only the viewing angles are allowed to vary, as a tripod mounted camera would, which is a less generic assumption than our models permit.

- Dynamic texture (Doretto et al., 2003)

  Learning maximum likelihood models of dynamic textures.

These are restricted to time series, which is less generic than our proposed models. The models are not trying to be faithful exact representations of the time series, instead they capture the "essence". These models are however intended to extrapolate this "essence", which we do not do. This is a different problem than the one we are trying to solve.

- Time series prediction by chaotic modelling of nonlinear dynamical systems (Basharat and Shah, 2009)

  Input images are embedded in phase-space. In this space new predictions are made, before transforming the result of these predictions back to image space.

  Our proposed models are more generic than this — they try to make no assumptions about the underlying system that causes images to be observed. We do not assume that there is any phase space or other intermediate space in which modelling makes any more sense, other than image space itself which facilitates this generic approach. As well as assuming there is an underlying dynamical system this approach also assumes that the input is a time series, thus time (frame number) is the only parameter controlling the input and synthesis of new output. In our models the output can be controlled by (almost) any number of parameters, (e.g. time, position, illumination) as directly defined by the sampling parameter themselves.

- Higher order SVD analysis for dynamic texture synthesis (Costantini et al., 2008)

  Higher order SVD is performed and used for dimensionality reduction by discarding some of the results. A multivariate first order autoregressive model is applied to learn the dynamics of the system in this reduced space. New texture is then synthesised based upon the results of this and some random noise.

  This system does not faithfully model the input sequence (some of it is

discarded) and furthermore the objective of the model is to synthesis other, *similar*, plausible sequences. The system is based on the assumption that the input is 'texture' and time is the only parameter exposed to control the output. Whilst it might be possible to extend this such that more than just time and noise to control the output the authors have not discussed this in the paper.

- Analysis and synthesis of textured motion: particles and waves (Wang and Zhu, 2002, 2004)

  Particles and waves are modelled as linear combinations of Laplacian of Gaussians (LoG) and Fourier series bases respectively. From this, parameters of the input video are learnt in order that new videos may be synthesised. Compared to our models this is constrained by the selection of bases (they may not be suitable for all/many data sources) and the target is synthesis of only plausible, not exact sequences. The sequences are further assumed to be only time series and not parametrised by anything else.

- A parametric texture model based on joint statistics of complex wavelet coefficients (Portilla and Simoncelli, 2000)

  A constrained statistical model of texture is built based upon a set of basis functions and images are decomposed based upon a set of linear filters.
  The aim of this is not to exactly represent a given image or set of images, but to instead facilitate the synthesis of plausibly similar other textures. The method is limited to a specific class of textures, and the parametrisation is inferred and based upon the set of filters selected — not directly by the input itself.

- Texture mixing and texture movie synthesis using statistical learning (Bar-Joseph et al., 2001)

  Statistical learning is used to learn a hierarchical multi-resolution analysis

tree from input samples. From this new random trees are created, by merging the trees observed in the input. This can be used to synthesise a new texture with similar statistics to the input(s).

This is not parametrised in such a way as to make it an accurate representation of an image manifold; the constructed trees represent individual images. This means that it does not offer a feasible framework in which to parametrically model an image manifold. It further assumes that the input images fall into the category of textures, which limits the appropriate inputs compared to our models. Movies synthesised in this way may not even always exhibit spatial and temporal coherence.

- Texture synthesis and modification with a patch-valued wavelet transform (Peyré, 2007, 2008)

  This work uses nonlinear manifold learning to perform texture synthesis, using one example texture.

  The manifold that is learnt is based upon *patches* of a single texture, which is a significantly different definition than our approach which uses multiple, whole images. Our method is more generic (we model arbitrary data, with unlimited parameters). This work synthesises other, novel, similar textures and does not provide a model of the manifold in the sense that we do and our model does not aim to synthesise other plausible textures so no direct comparison is possible.

- Transferring colours to grayscale images by LLE (Li et al., 2008)

  The authors propose a semi-automated method for adding (plausible) colours to grayscale images by using LLE on patches of (manually selected) similar images.

  Our models do not add colour. The learning method employed is LLE, which is discussed separately. The meaning of "image manifold" in this context is not the same as ours, and the synthesis is extrapolation, not interpolation.

Direct comparison is not possible because of these differences.

- Image Manifold interpolation using free-form deformations (Souvenir et al., 2006)

  The authors propose to use nonlinear manifold learning to discover the parametrisation of heart images (both breath and pulse cycles contribute to its deformation). From this they use B-Splines to model the *deformation* (as fields) present in images, across all possible images.

  This work assumes images are of objects which are only being distorted in the image plane itself. The B-Spline modelling is performed across two layers (i.e. there are two B-Spline surface models constructed, with one for position in an image and the second for position on the manifold) which is wasteful for only a 2-D parametrisation. The authors do not consider cases where there are more than two parameters controlling the deformation. For these reasons this work is much less generally applicable than our methods. It is interesting to note however that in their evaluation the authors use a very similar approach to our approach.

A.3 Appearance based vision (general)

- Subspace methods for robot vision (Nayar et al., 1996)

  The authors propose the use of PCA for robot vision problems, using images themselves as input instead of feature detection.

  The main contribution of this paper is the PCA based approach to robot control. This is effectively parameter learning, and synthesising new images in this way would be substantially different to our methods.

- Rotation invariant appearance based maps for robot navigation (Neal and Labrosse, 2004)

  Topological maps of an environment are constructed based upon distances

in image space.

This does not provide a feasible method for synthesising new images. It would however make for a sensible sampling strategy for guiding the acquisition of sufficient samples of an image manifold, for use by another algorithm.

- Visual homing based on Fourier transformed images (Sturzl and Mallot, 2006)

  Sub-symbolic visual homing is performed, however instead of conducting this in image space it is performed in frequency space.

  This may be an interesting future line of inquiry, however modelling image manifolds in this space is beyond the scope of this thesis.

- Segmenting cardiopulmonary images using manifold learning (Zhang and Pless, 2005; Zhang et al., 2006)

  This work applies manifold learning techniques to develop additional constraints for segmentation of cardiac images.

  The manifold learning applied is relatively standard (a distance metric based on Gabor filters is used however). The paper only looks at cardiac data and this is only considered as a 2-D dataset.

- Voronoi-based segmentation of cells on image manifolds (Jones et al., 2005)

  Riemannian metric for segmentation of cells.

  The authors use a different concept of image manifolds than we are using in our work, choosing to operate in the image plane primarily instead of image space as we do.

- Motion recovery by integrating over the joint image manifold (Goshen et al., 2005)

  This paper introduces the concept of the joint image manifold as a method for processing stereo pairs to recover motion.

  Several fitting methods are proposed. This problem area is substantially

different than what we look at in this thesis. We do not consider motion recovery, and we do not use any stereo datasets.

- Tracking people on a torus (Elgammal and Lee, 2009)

  This work uses manifold learning to connect images and pose data with the aim of estimating posture from inputs.

  The manifold learning is supervised and, as with other manifold learning, unrelated. Additionally the manifold is assumed to be topologically equivalent to a torus, which is somewhat less generic than our models.

- 3D body pose from silhouettes using manifold learning (Elgammal, 2004; Elgammal and Lee, 2008, 2004, 2007)

  Manifold learning (LLE) is used to construct a mapping between views, a low dimensional embedding and 3D pose.

  The relevant areas of this work to ours are still solving a different problem, and the reasons discussed with LLE apply here.

- Learning a manifold-constrained map between image sets (Ham et al., 2006)

  Manifold learning is employed and constrained in such a way that a direct map between two image manifolds is learnt. The approach is semi-supervised in that a small number of correspondences are given.

  This is a substantially different problem because of the mapping between manifolds being constructed, and direct comparison to our methods is not possible because of this. We know the parametrisation of our datasets, which means we would have 100% of the correspondences available, however this doesn't enable us to conduct experiments under the same framework used to evaluate our own models. This paper is however important because it explicitly states a number of the observations about manifold learning algorithms that motivate our method.

- Learning shared latent structure for image synthesis and robotic imitation

(Shon et al., 2005)

(This is probably better described as IBR, but follows from the previous paper.) This work uses Gaussian process regression to link two different sets of observations (images) via a low dimensional latent variable space.

As with (Ham et al., 2006) this work is looking at directly connecting two sets of inputs. New views are synthesised that weren't in the training set, but these are produced based on input views in the other dataset. The latent variable space does not reflect the parameters that were used to acquire the input images. We can't use this to construct a fair comparison to our methods.

A.4 Faces and lips

- Eigenfaces (Turk and Pentland, 1991*a*,*b*)

  By performing PCA on sets of face images and projecting into a lower dimensional space than the image space itself, face recognition may be performed by nearest neighbour searching.

  See subspace methods for robot vision entry for why PCA is not a direct competitor to our work.

- Multi-view face recognition by nonlinear tensor decomposition (Tian et al., 2008)

  HOSVD is employed as a dimensionality reduction technique. This is used as the basis of a representation which permits matching new, unseen views about previously learnt ones.

  As with other manifold learning approaches this is not suitable for direct comparison with our methods. The remainder of this paper focuses on estimating identity and view.

- Face recognition with image sets using manifold density divergence (Arand-

jelovic et al., 2005)

This work learns probability densities, constrained to a low dimensional manifold for recognition.

The learning method and model proposed are not suitable for direct comparison to our models because of the learnt parametrisation. The model does not exactly represent sampled images either.

- Analysis and synthesis of lip images using dimensionality reduction (Aharon and Kimmel, 2006)

  Manifold learning is conducted using LLE, MDS and variants. Images are recognised based on distances with the learnt embedding. New speech is also synthesised, formulated as a traversal of this embedding.

  The synthesis element of this work only uses this to select appropriate, previously seen images in a plausible order, using graph traversal algorithms. LLE, etc., are discussed separately.

- Manifold based analysis of facial expressions (Hu et al., 2004)

  This work focuses heavily on the learning aspects. Facial expressions and transitions between them are learnt using manifold learning techniques such that they may be recognised or synthesised.

  This work is performed in a reduced space, not full images space as our models do. This is based upon feature detection, which makes it somewhat domain specific and less generic than our proposed methods. Furthermore the learning of parameters makes it unsuitable for direct comparisons with our models.

- Facial expression synthesis using manifold learning (Huang and Su, 2006; Su and Huang, 2005)

  Dimensionality reduction, combined with a Markov random field is used to infer new facial expressions.

This is extrapolation, not interpolation, so comparison with our models are not sensible. The generated face images are patch based, with best matching patches stitched together to produce the output.

- Posed face image synthesis using nonlinear manifold learning (Cho et al., 2003)

  This work uses LLE as the basis of an image synthesis algorithm.

  Since LLE is used to learn the parametrisation of the manifold we cannot compare this directly to our own methods where the known parametrisation is used.

A.5 Manifold learning/Dimensionality reduction

- ISOMAP (Tenenbaum et al., 2000)

  Generates a low dimensional embedding using geodesic distances.

  This is solving a different, but related problem — we already know the parametrisation of our datasets at the outset. This means that the parametrisation we could learn will quite likely not be exactly representative of the parametrisation we inherently knew already. The net effect of this is that if we want to synthesise an image at $u = 5, v = 2$ we cannot just use the embedding generated by ISOMAP to do this, because it does not map directly. Furthermore the ISOMAP procedure does not use all of the information (see the residual variance), where as our models attempt to faithfully recreate the input images.

- LLE (Roweis and Saul, 2000)

  LLE is similar to ISOMAP, except that instead of using geodesic distances weights are computed such that each point is represented by a linear combination of its neighbours. Note that this is local, not global as ISOMAP was. As with ISOMAP this is finding new parametrisations of data we already

have a known parametrisation for and then discarding some of this in order to achieve a reduction in dimensionality.

- Laplacian Eigenmaps (Belkin and Niyogi, 2003)

  Spectral graph theoretic approach to manifold learning.

  Again this method 'discovers' parametrisation for something we already have parameters for. As with previous manifold learning methods the solution involves taking m of k solutions to an eigenvalue problem. As previously the non-linearity comes from the parameter learning stage, not the eigen decomposition.

- GTM (Bishop et al., 1996, 1998*a*,*b*)

  GTM seeks to learn, via expectation maximization, a gaussian mixture which models the probability distribution that generated the original data.

  The dimensionality of the hidden variable space can clearly be selected to be the same as the dimensionality of the parameter space of the data. However this does not mean that the parametrisation of the latent space itself will be reflective of the (known) parametrisation of the data. Better noise modelling would require more data samples than are available with our datasets.

- General dimensionality reduction/manifold learning (Robert and Richard, 2009; van der Maaten et al., 2009)

  In addition to ISOMAP, LLE, etc., a great many other manifold learning algorithms have been proposed. These papers review and classify a number of these.

  The reasons why these techniques are not directly the same as the models presented in this thesis are the same as for the afore mentioned manifold learning techniques.

- Estimating manifold dimension by inversion error (Martin and Bäcker, 2005)

  The authors measure the error when inverting (i.e. regenerating) the original

images after performing dimensionality reduction to determine if the dimensionality reduction has reduced the dataset sufficiently or too much.

For the purposes of our work this introduces nothing notable over standard LLE and ISOMAP procedures, which we have discussed previously.

- Learning nonlinear image manifolds by global alignment of local linear models (Verbeek, 2006)

  The author proposes a global, two-way mapping based upon combinations of small scale local mappings.

  This learns the parametrisation of image manifolds. The fact that the parametrisation is learnt, rather than specified, means that it is not possible to construct a fair comparison with our models, which are intended to reflect a known parametrisation.

- Learning to traverse image manifolds (Dollár et al., 2006)

  An iterative minimisation procedure is used with radial basis functions to model image manifolds.

  As with other manifold learning techniques the learnt parametrisation does not necessarily reflect the (known) real parametrisations of our datasets. This in turn makes fair comparisons to our models impractical. In this work, when image manifolds are modelled they are split into patches, which are modelled separately.

- Nonlinear image interpolation using manifold learning (Bregler and Omohundro, 1994)

  The authors propose to learn image manifolds as local linear patches "glued" together. Several interpolation approaches are proposed, which constrain the generated image to fall on the modelled manifold.

  As with other manifold learning approaches the learnt parameters do not reflect the known parameters which makes direct comparisons with our models

infeasible.

- Regression on manifolds using kernel dimension reduction (Nilsson et al., 2007)

  Introduces a supervised manifold learning algorithm based upon Laplacian eigenmaps.

  This is not directly competing with our models for the reasons outlined previously for Laplacian eigenmaps in addition to the undesirable nature of any supervision introduced into the process.

- Semi-supervised dimensionaility reduction using pairwise equivalence constraints (Cevikalp et al., 2008)

  Integrates equivalence constraints into dimensionality reduction procedure in order to facilitate labelling of different classes.

  We don't have a concept of 'different classes' in our models. We assume that all images in a given dataset come from the same object or environment.

- Image retrieval using nonlinear manifold embedding (Cai et al., 2007; He et al., 2004; Wang et al., 2009)

  Image retrieval aims to bridge the semantic gap between high level concepts and low level features. Several authors recently have proposed non linear manifold learning algorithms as a solution to this.

  This is a semi-supervised manifold learning process. It is unsuitable because of the semi-supervised labelling phase, as well as the more general non-applicability of manifold learning techniques to our work.

- Non-linear CCA and PCA by Alignment of local models (Verbeek et al., 2004)

  This work looks at cases where there are multiple low-dimensional embeddings.

  In our case we already know the embedding we are interested in and this is

the only one which is of use to us. In these cases this method reduces to be equivalent to Laplacian eigenmaps.

- Finding minimal parameterizations of cylindrical image manifolds (Dixon et al., 2006)

  This paper looks at solving the manifold learning problem for known cylindrical manifolds.

  As with other previous manifold learning techniques this is solving a different problem than we solve with our models. Additionally our models are more generic than this in that they consider broader cases than just two dimensional cylinders.

- Image distance functions for manifold learning (Souvenir and Pless, 2007)

  This paper looks at a specific domain for manifold learning (videos) and applies ISOMAP, in addition to applying standard ISOMAP they consider several alternative distance measures which may be more applicable to specific domains.

  Our use of distance metrics is somewhat different (where this work uses distance measures to control the ISOMAP procedure we use them to compare synthesised and known images). The arguments for not using ISOMAP and manifold learning apply here.

- Rank-R approximation of tensors using image-as-matrix representation (Wang and Ahuja, 2005)

  The main contribution of this paper is an efficient, novel algorithm for generating such approximations.

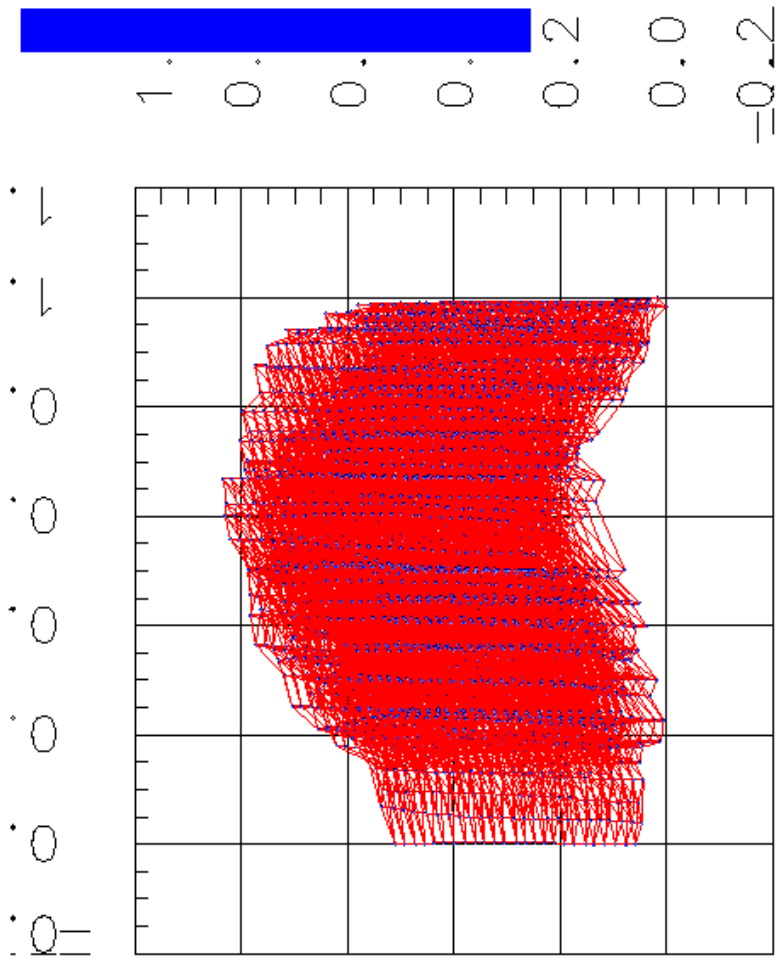
  The paper focuses on video sequences, and the algorithm does not support a link between the (known) acquisition parameters of the images which makes direct comparison to our method impractical. The aim of this is the generation of reduced dimension approximations.

Appendix B

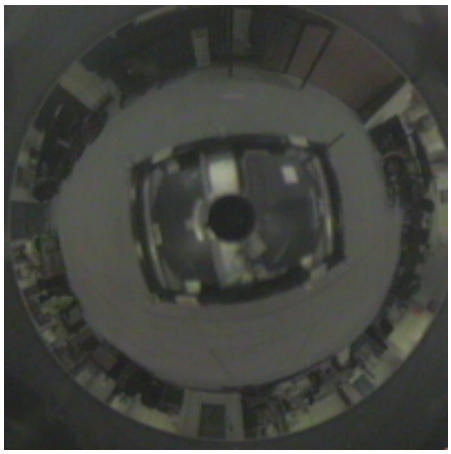
Image Manifold list

In this appendix we list the manifolds we studied, and provide sample images in order that the reader might better understand the type of images used. We also provide parameter space visualisations of the acquisition of the data in order that the reader may see how the parameter space was sampled. In the case of 1-D manifolds, or manifolds of dimension ≥ 4 this is not provided. In the former case these visualisations do not say anything and in the latter case it is not possible to usefully convey the information required through a fixed figure on a page.

For more details on these manifolds see Table 3.1, page 64.



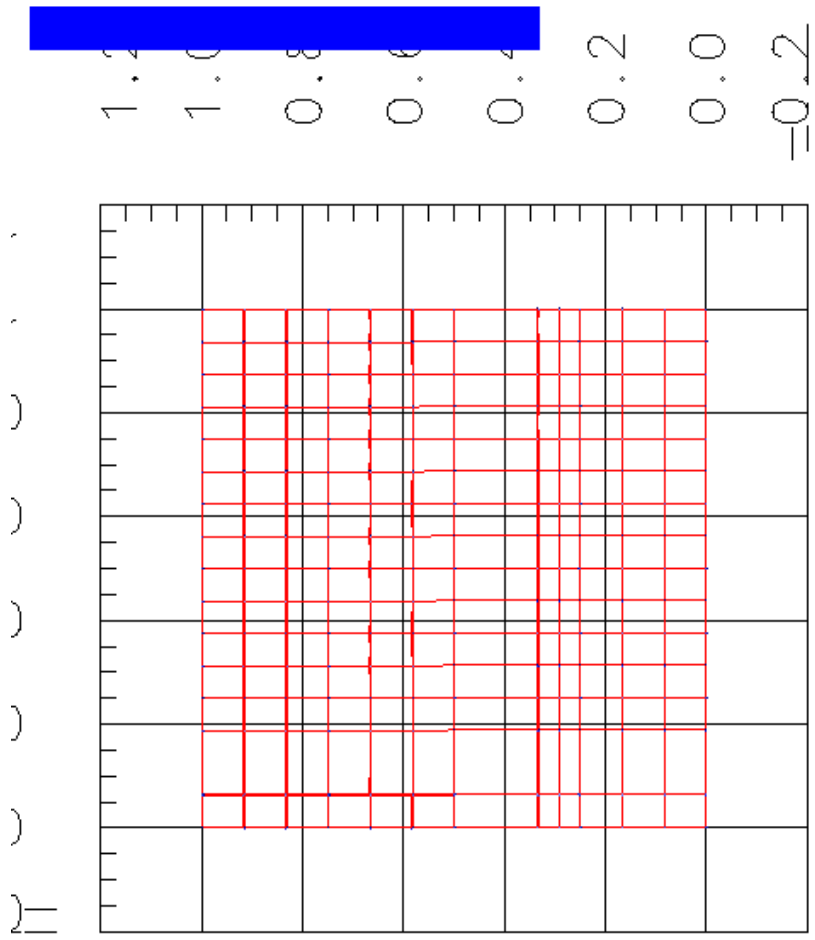
(a) Visualisation of parameter space of manifold EXPT03



(b) Sample image at $(52.4166, -4.06644, 4.25965)$

(c) Sample image at $(52.4166, -4.06645, 3.07706)$

Figure B.1: The manifold EXPT03

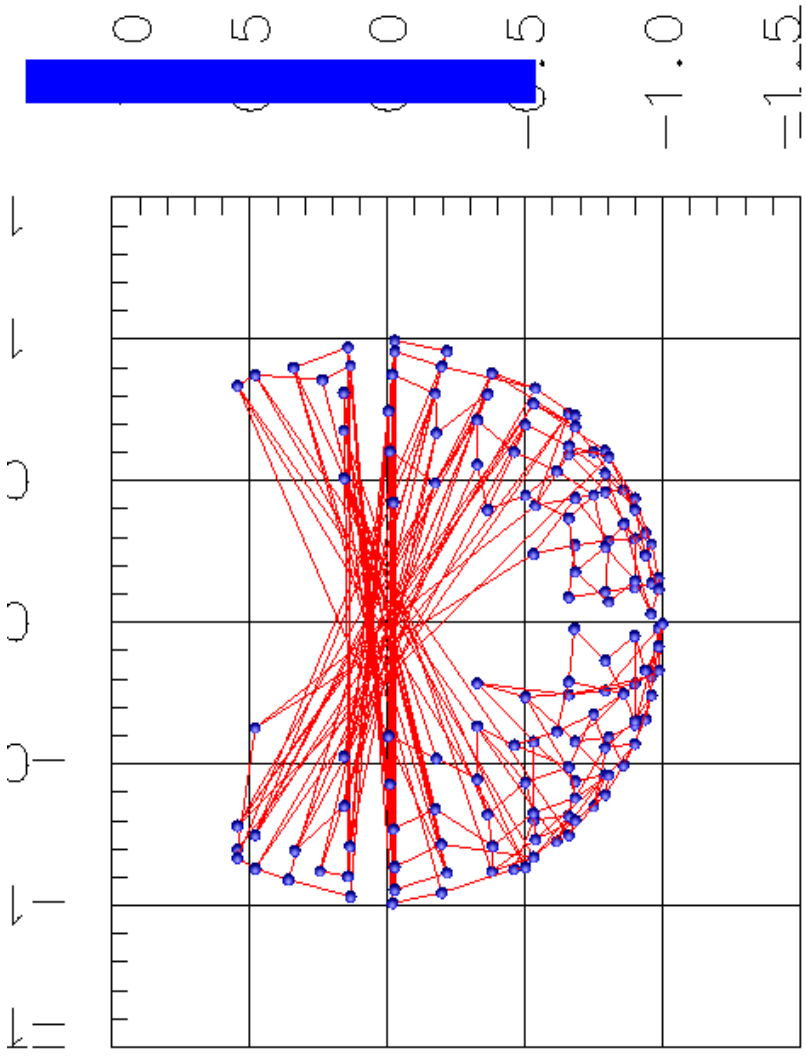


(a) Visualisation of parameter space of manifold WINDOW3

(b) Sample image at $(-6385, -3185, 1.22742e+09)$

(c) Sample image at $(6400, 1951, 1.22753e+09)$

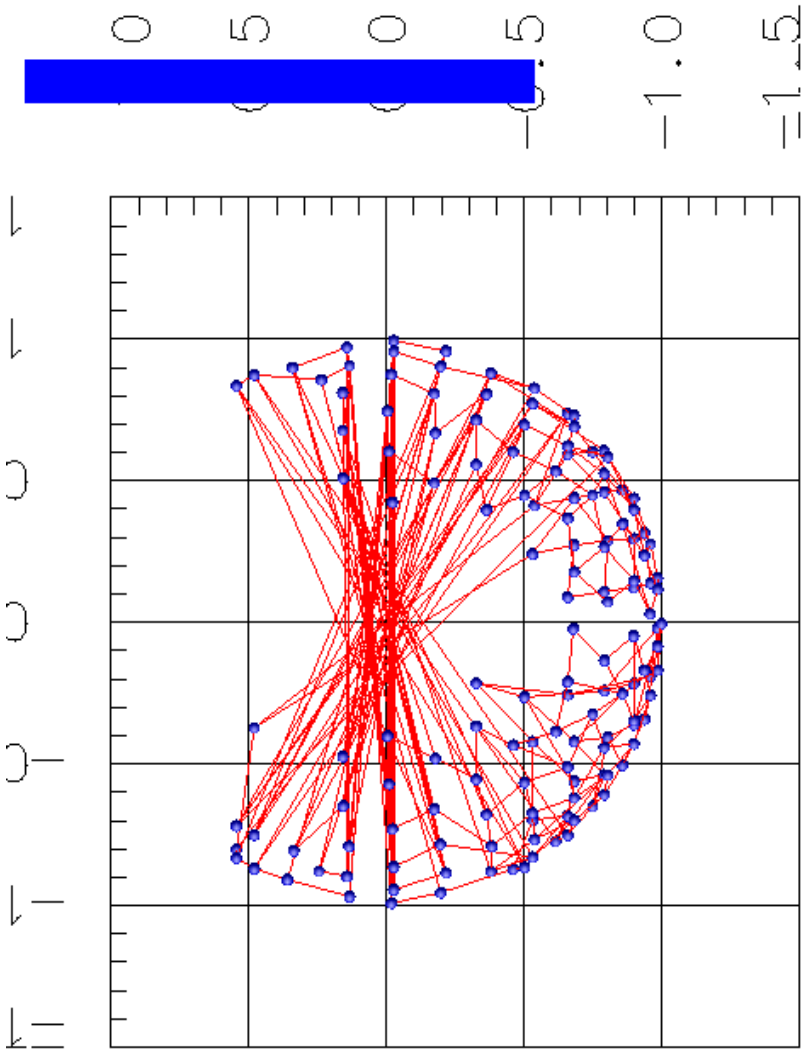
Figure B.2: The manifold WINDOW3



(a) Visualisation of parameter space of manifold KNIGHTFIGHTING

(b) Sample image at

(c) Sample image at

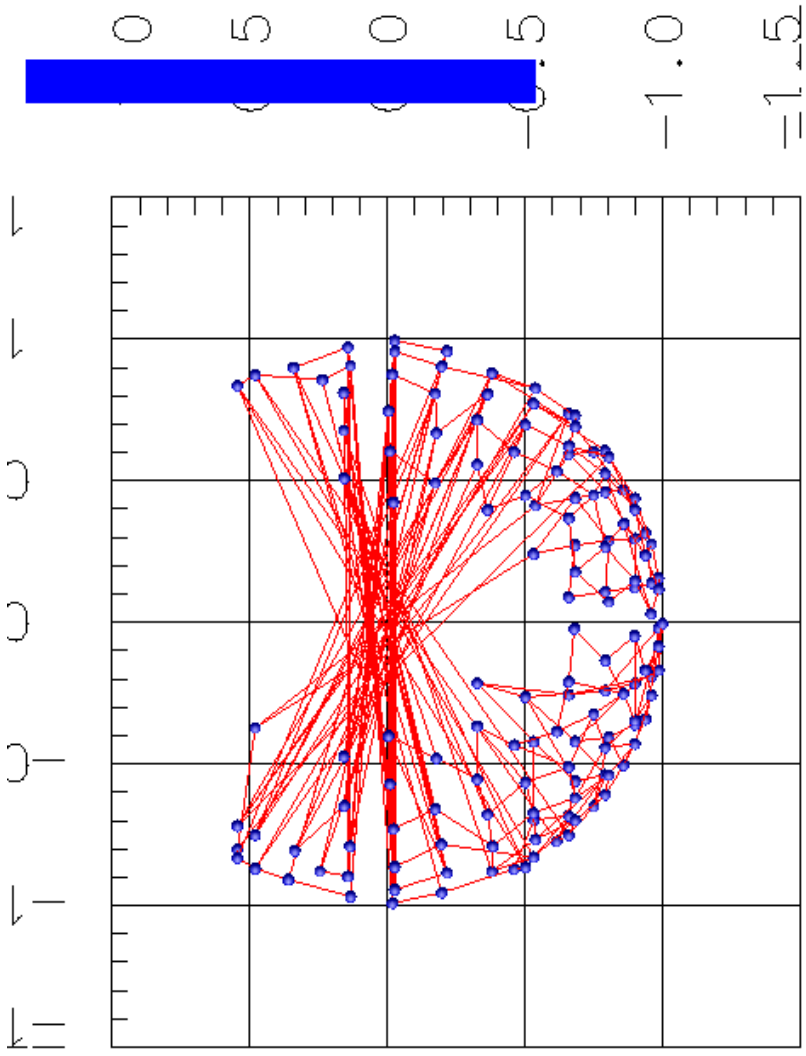


(a) Visualisation of parameter space of manifold KNIGHTKNEELING

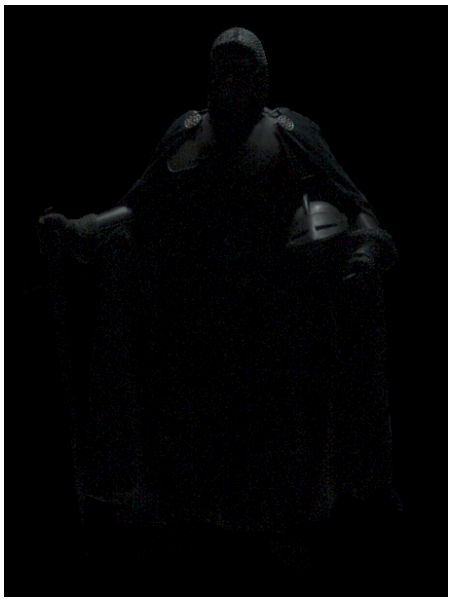
(b) Sample image at $(-0.084766, -0.985072, 0.149823)$

(c) Sample image at $(-0.014632, 0.545059, -0.83827)$

Figure B.4: The manifold KNIGHTKNEELING

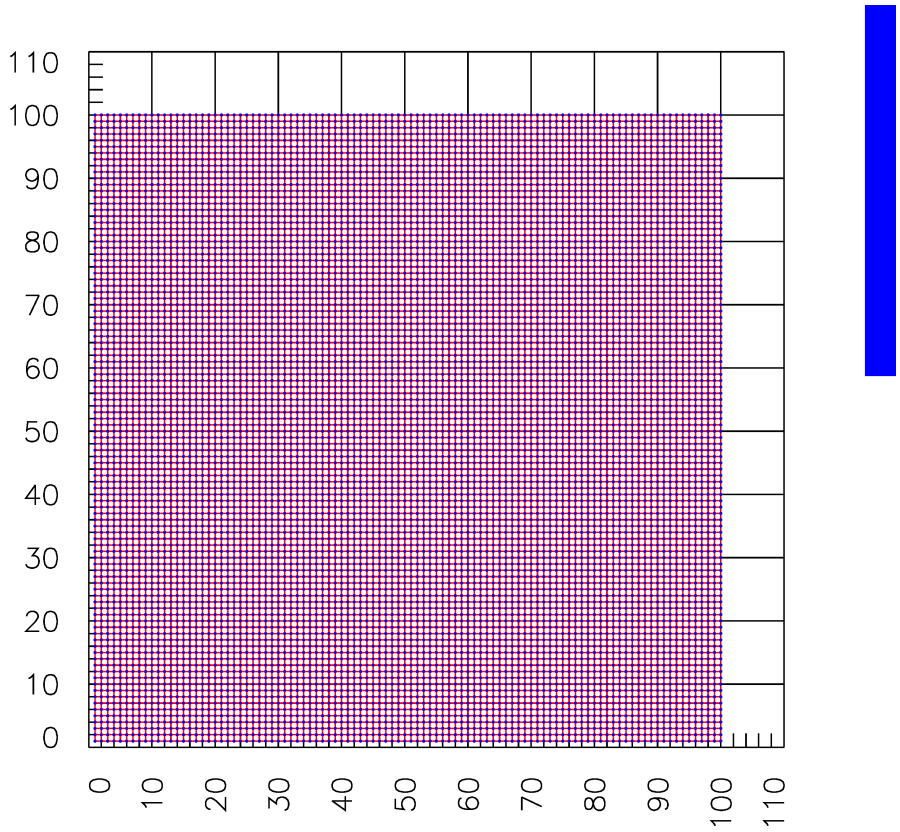


(a) Visualisation of parameter space of manifold KNIGHTSTANDING

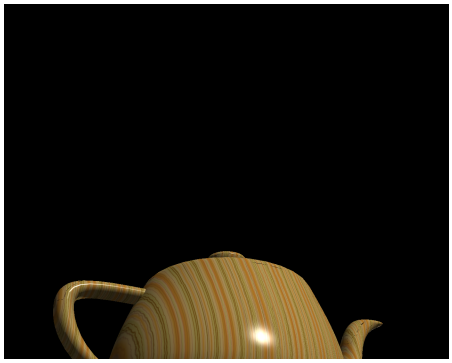


(b) Sample image at

(c) Sample image at



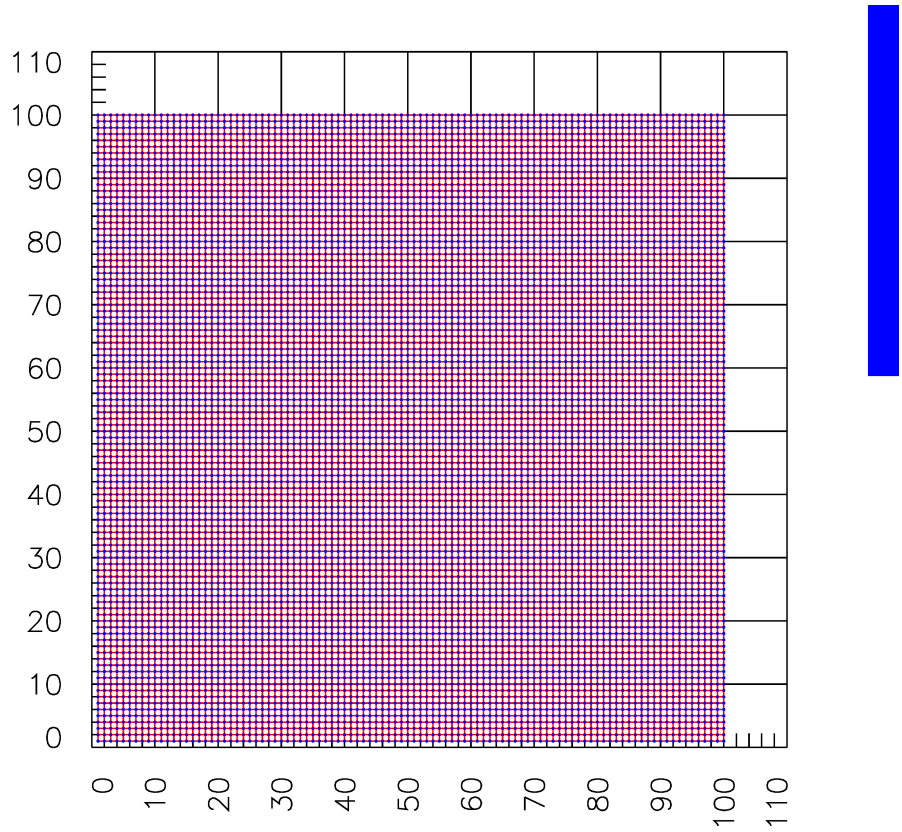
(a) Visualisation of parameter space of manifold 2DTEAPOT



(b) Sample image at $(1, 1)$

(c) Sample image at $(100, 98)$

Figure B.6: The manifold 2DTEAPOT

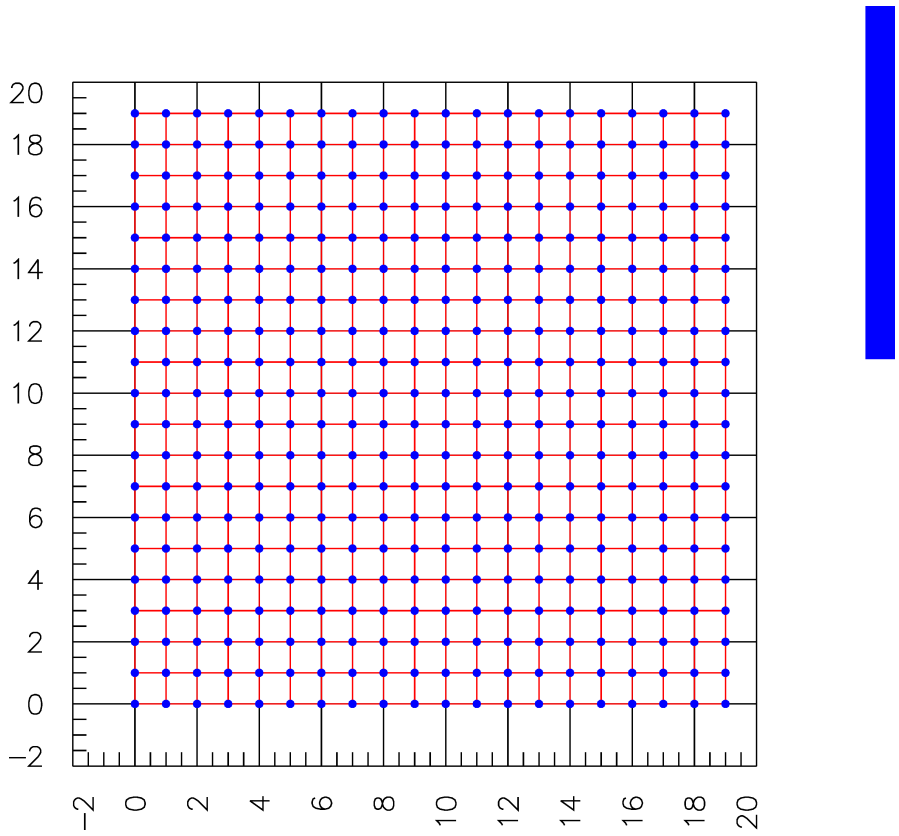


(a) Visualisation of parameter space of manifold 2DWOODBOX

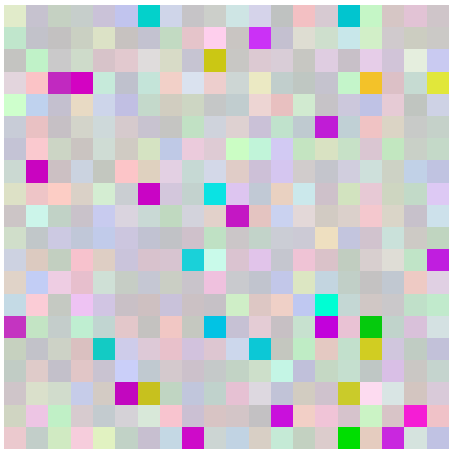
(b) Sample image at $(1, 1)$

(c) Sample image at $(100, 98)$

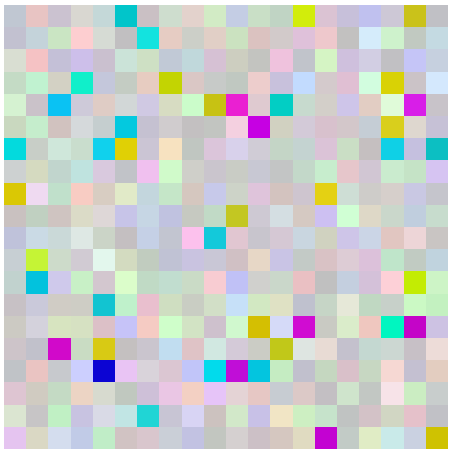
Figure B.7: The manifold 2DWOODBOX



(a) Visualisation of parameter space of manifold BMNOISE

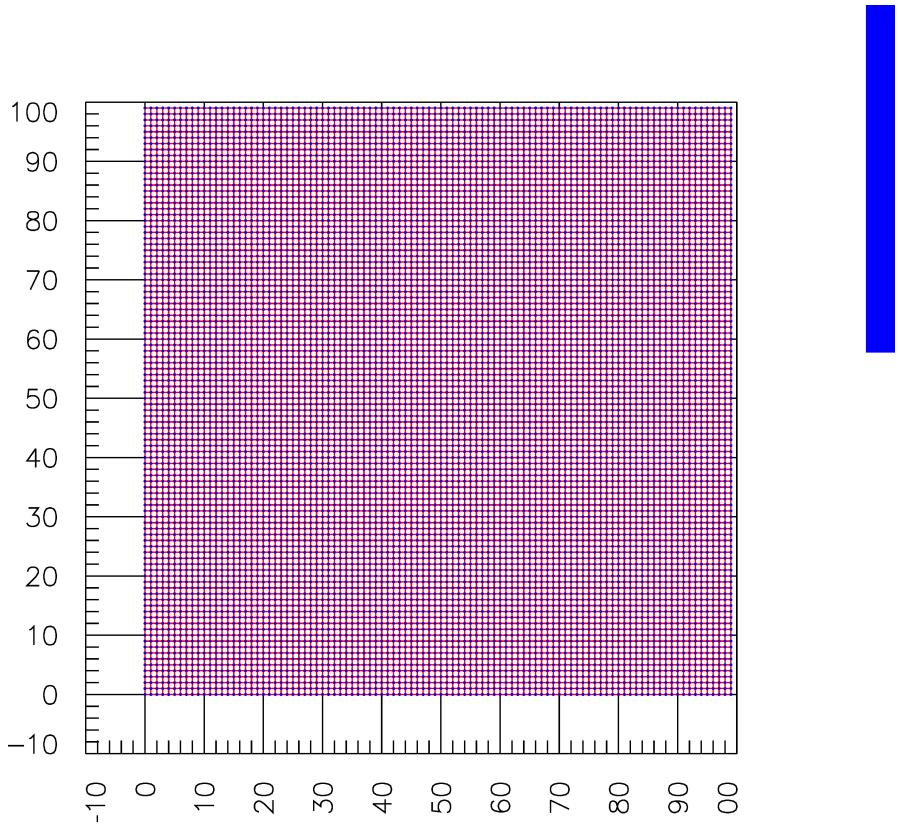


(b) Sample image at $(0, 0)$

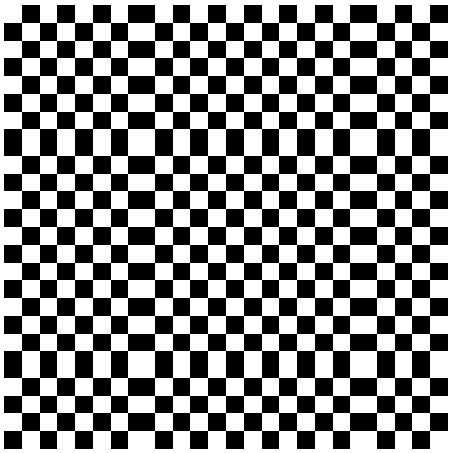


(c) Sample image at $(19, 17)$

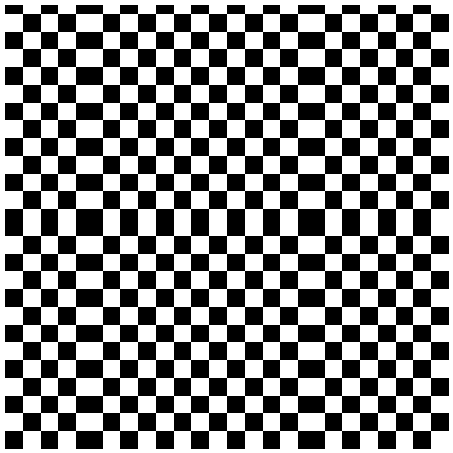
Figure B.8: The manifold BMNOISE



(a) Visualisation of parameter space of manifold CHESSBOARD

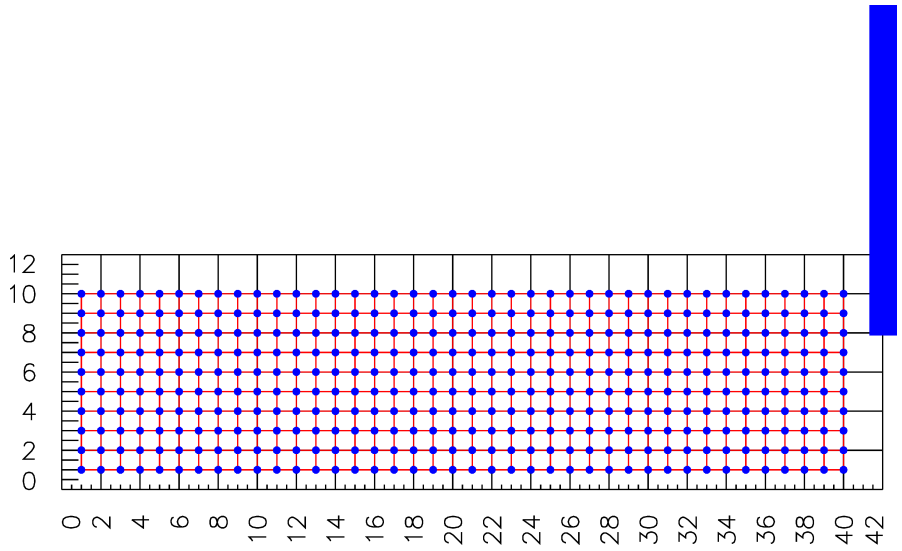


(b) Sample image at $(0, 0)$

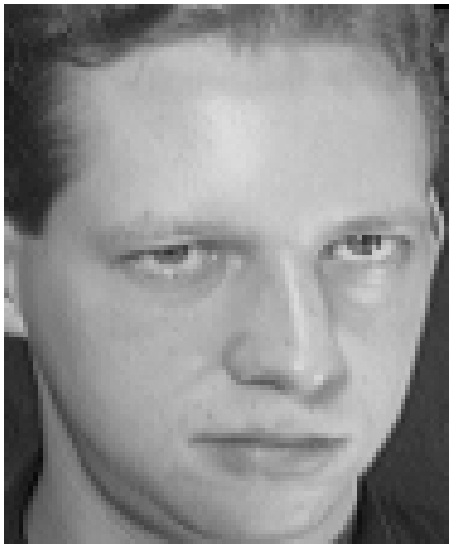


(c) Sample image at $(49, 47)$

Figure B.9: The manifold CHESSBOARD



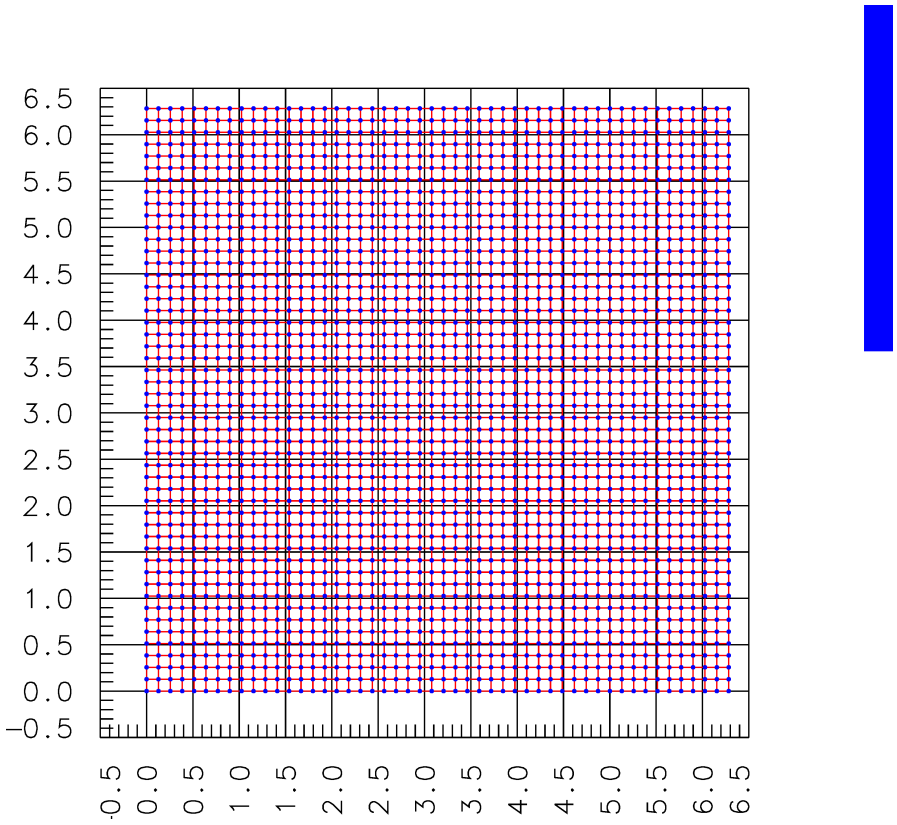
(a) Visualisation of parameter space of manifold FACES



(b) Sample image at $(1, 10)$

(c) Sample image at $(9, 7)$

Figure B.10: The manifold FACES

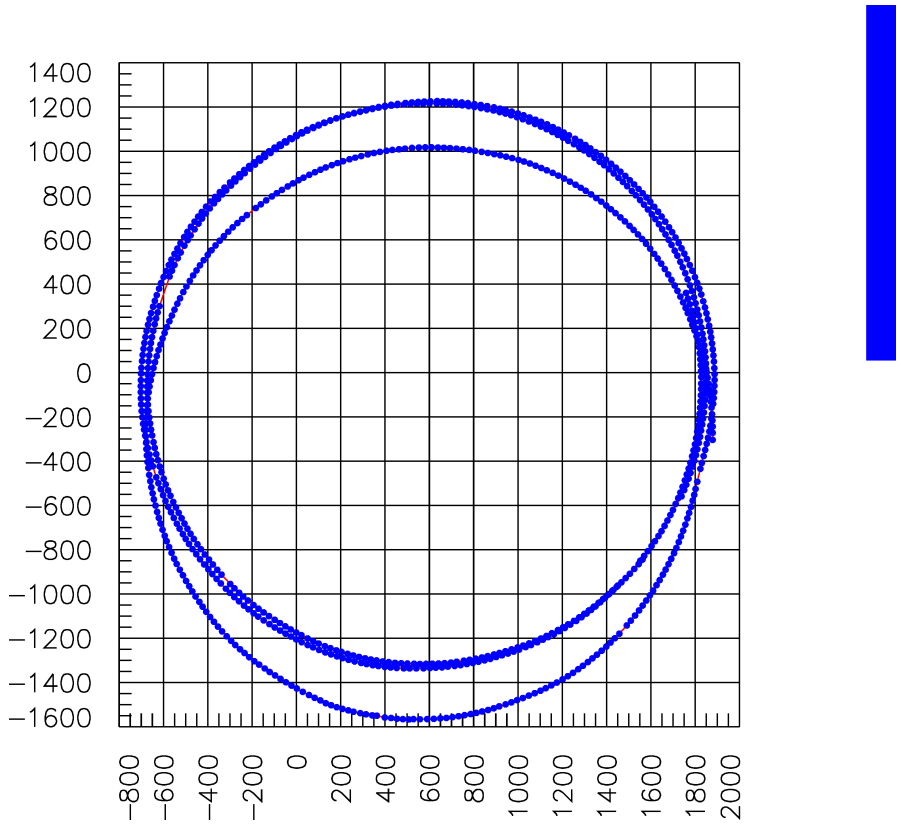


(a) Visualisation of parameter space of manifold SINECOS

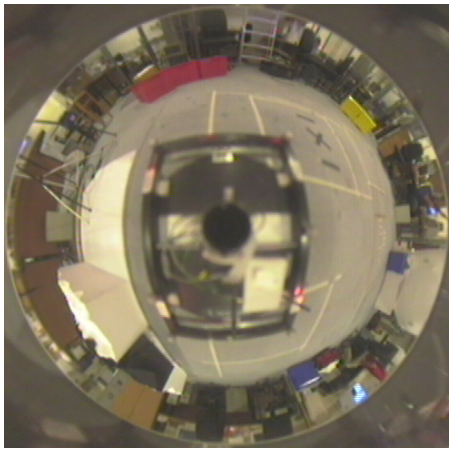
(b) Sample image at $(0, 0)$

(c) Sample image at $(6.28319, 6.02673)$

Figure B.11: The manifold SINECOS



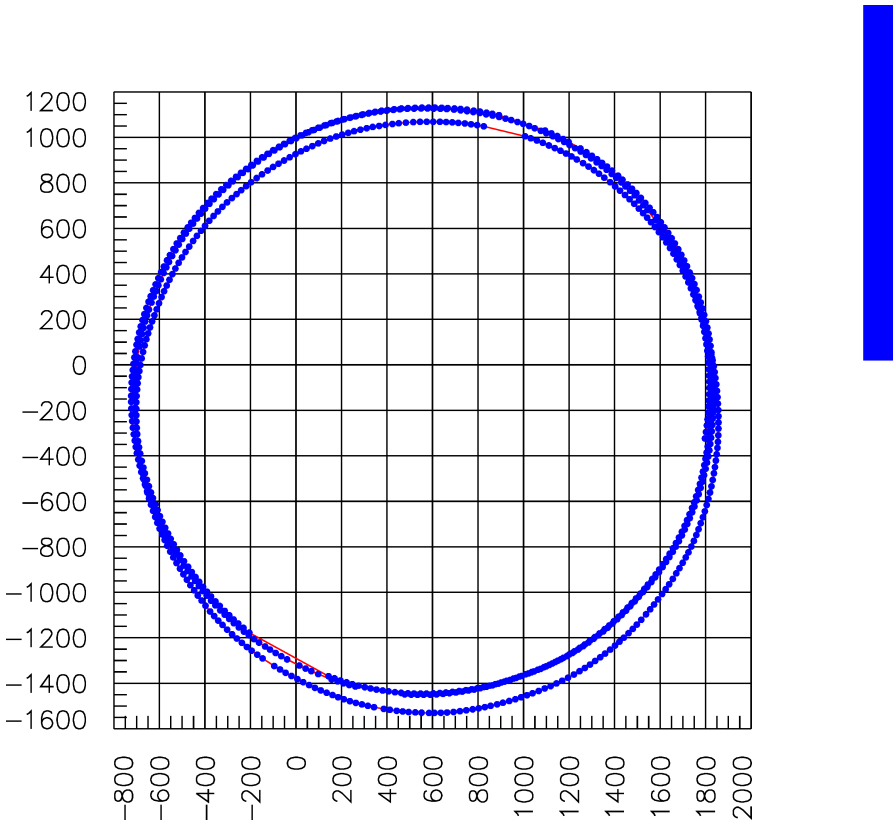
(a) Visualisation of parameter space of manifold CIRCLE3



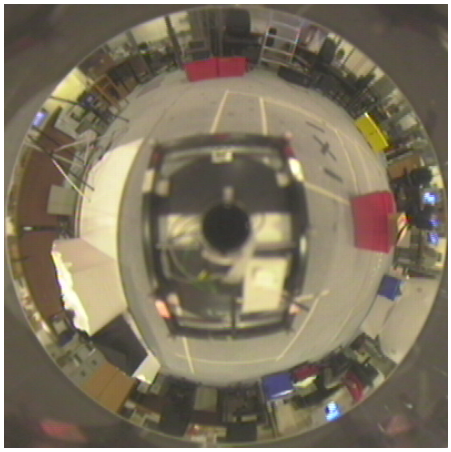
(b) Sample image at $(1878.77, -303.976)$

(c) Sample image at $(1775.2, 308.504)$

Figure B.12: The manifold CIRCLE3



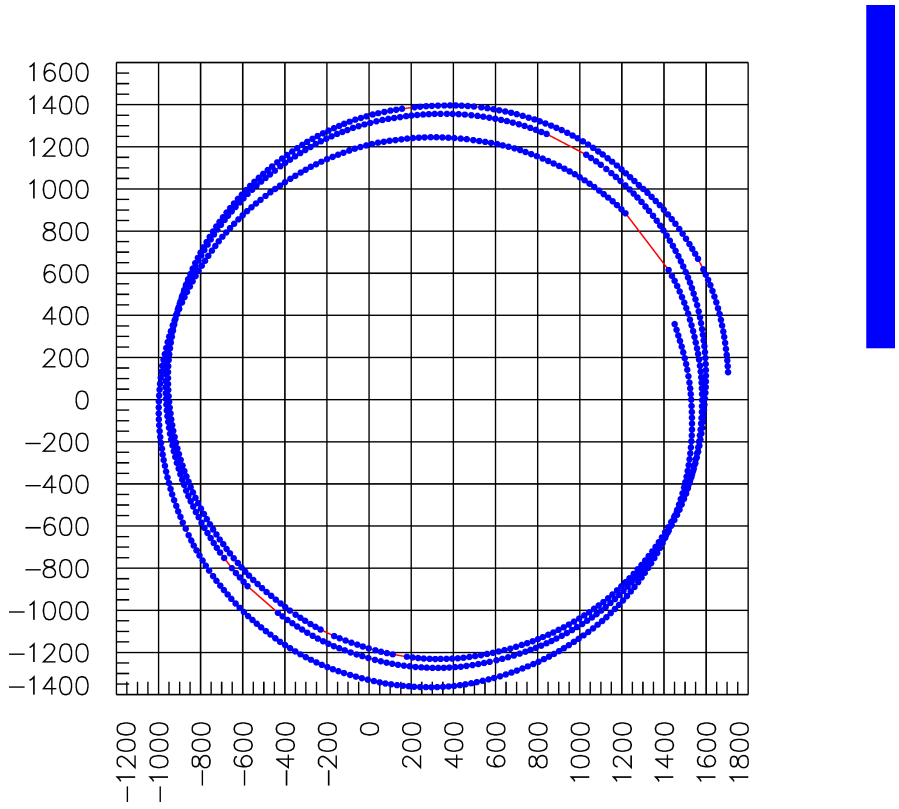
(a) Visualisation of parameter space of manifold CIRCLE4



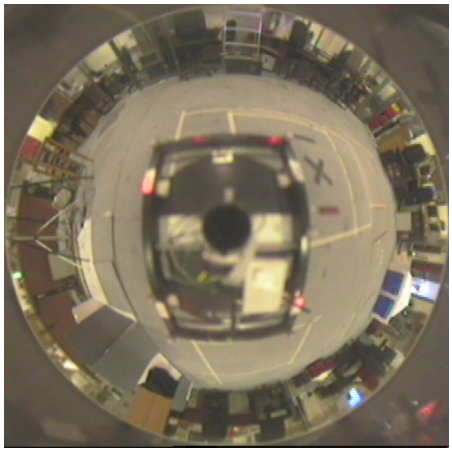
(b) Sample image at (1797.12, −323.729)

(c) Sample image at (1835.93, −6.32309)

Figure B.13: The manifold CIRCLE4



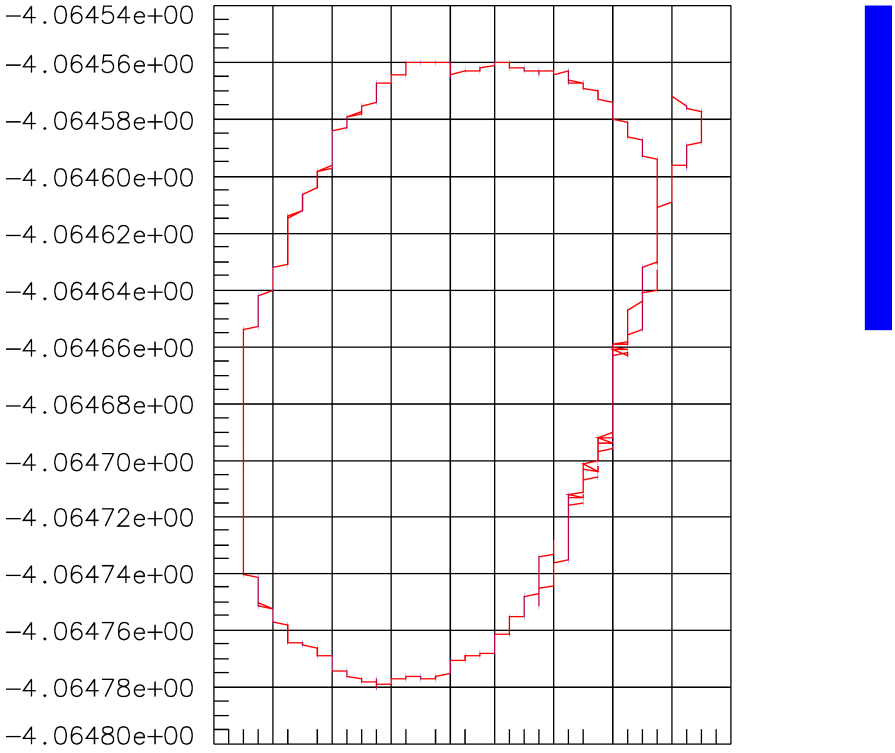
(a) Visualisation of parameter space of manifold CIRCLE5



(b) Sample image at $(1704.2, 130.272)$

(c) Sample image at $(1467.98, 304.46)$

Figure B.14: The manifold CIRCLE5

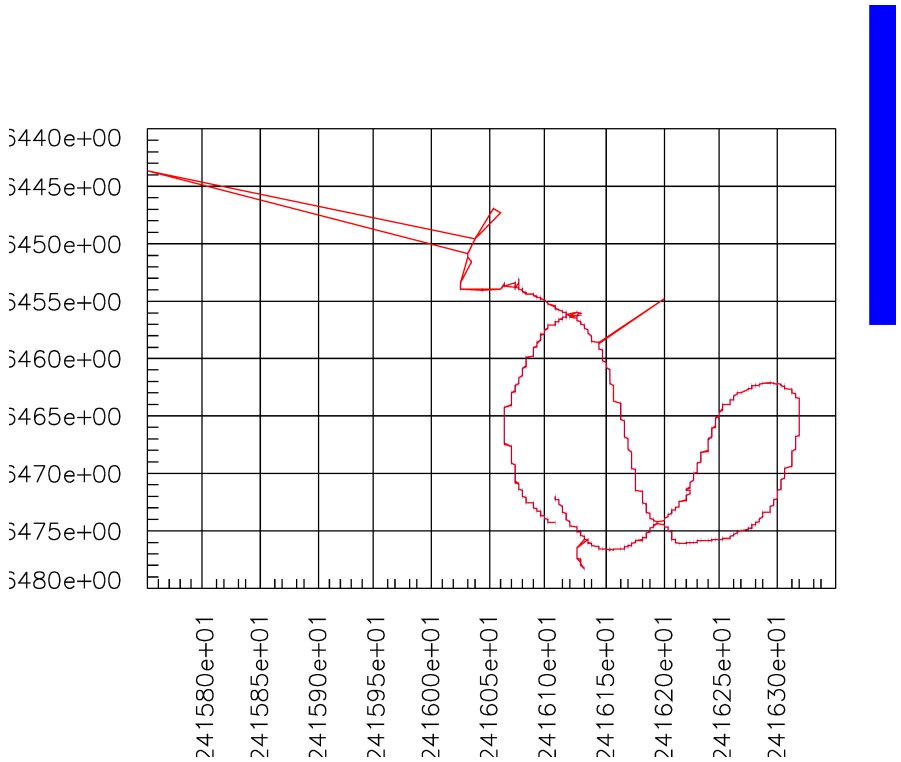


(a) Visualisation of parameter space of manifold IDRISCIRCLE

(b) Sample image at (52.4162, −4.06457)

(c) Sample image at (52.4162, −4.06475)

Figure B.15: The manifold IDRISCIRCLE

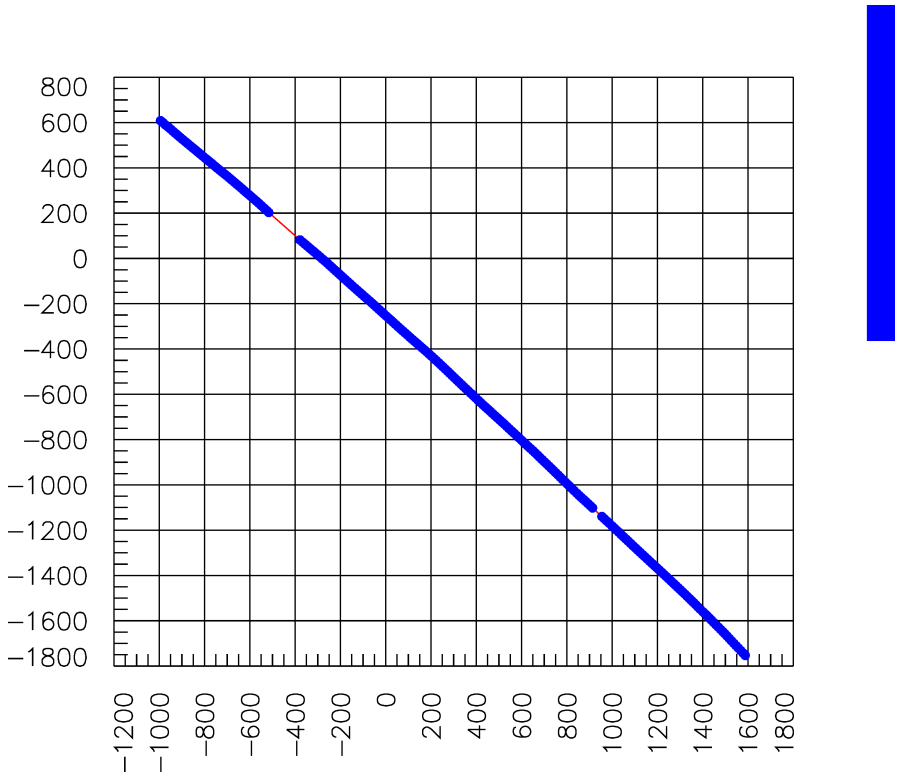


(a) Visualisation of parameter space of manifold IDRISFIG8

(b) Sample image at (52.4161, −4.06478)

(c) Sample image at (52.4161, −4.06472)

Figure B.16: The manifold IDRISFIG8

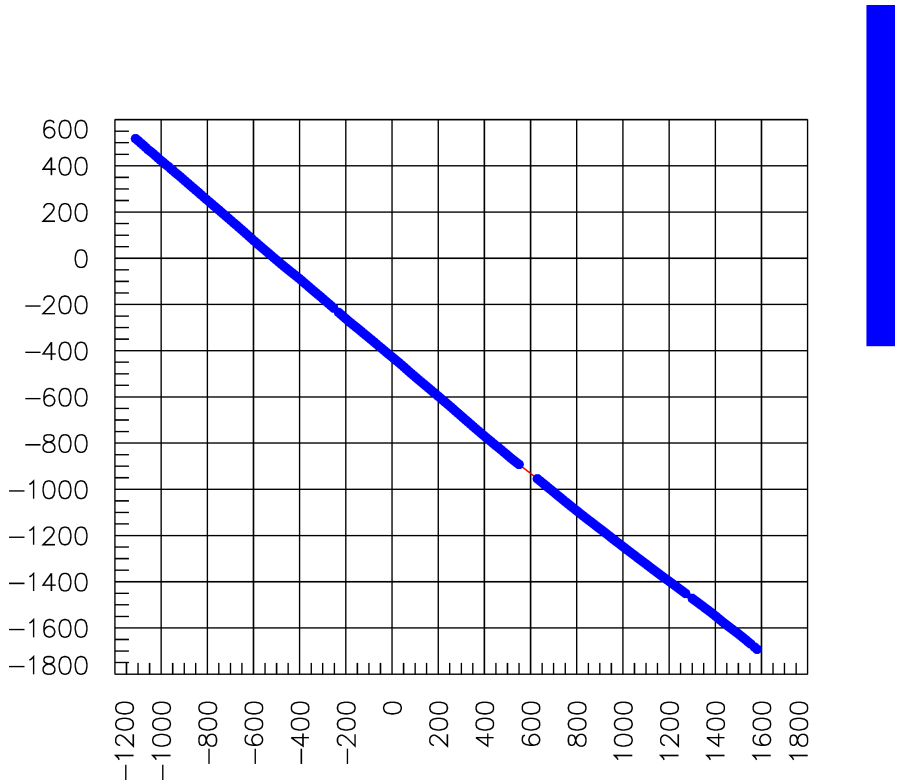


(a) Visualisation of parameter space of manifold STRAIGHT1

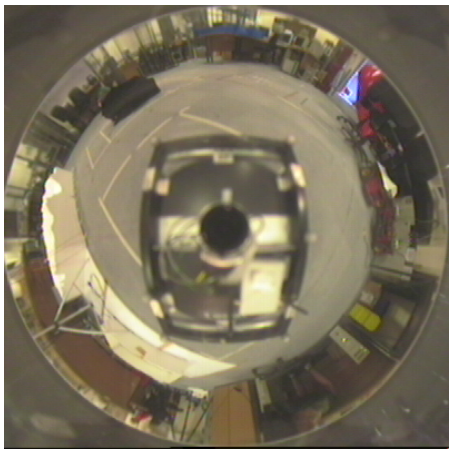
(b) Sample image at $(1587.75, -1753.04)$

(c) Sample image at $(-972.355, 589.219)$

Figure B.17: The manifold STRAIGHT1



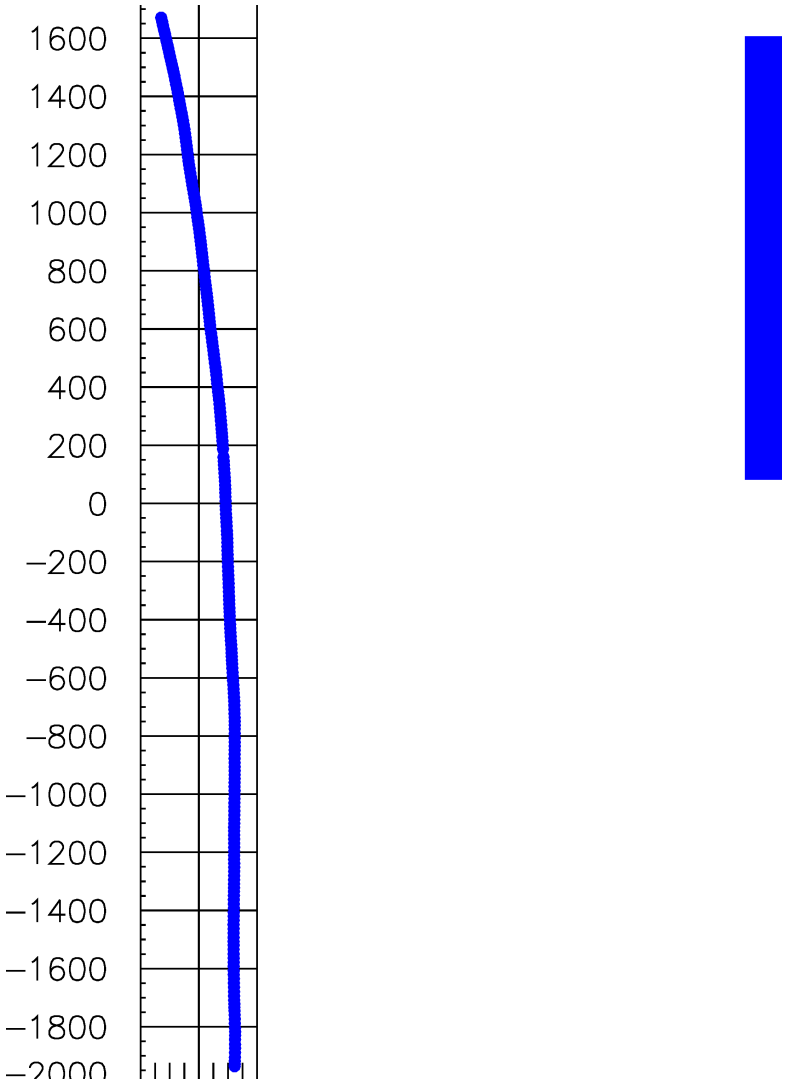
(a) Visualisation of parameter space of manifold STRAIGHT2



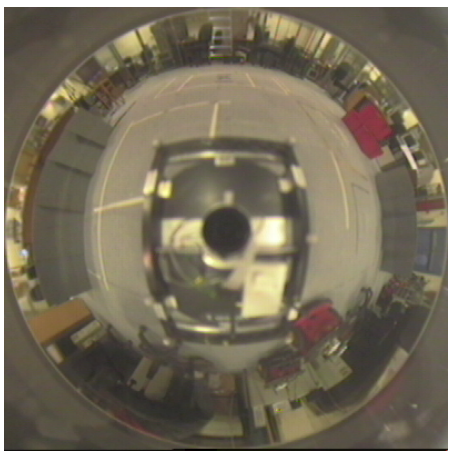
(b) Sample image at $(1580.93, -1692.34)$

(c) Sample image at $(-1087.82, 498.275)$

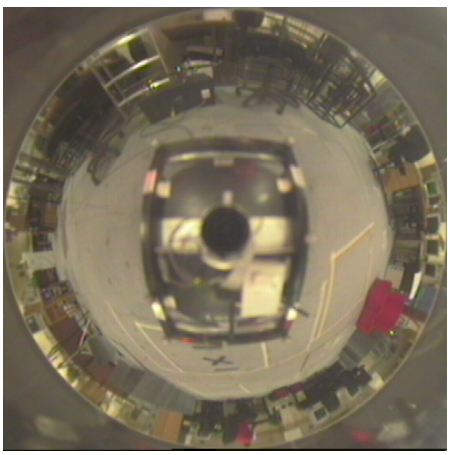
Figure B.18: The manifold STRAIGHT2



(a) Visualisation of parameter space of manifold STRAIGHT3

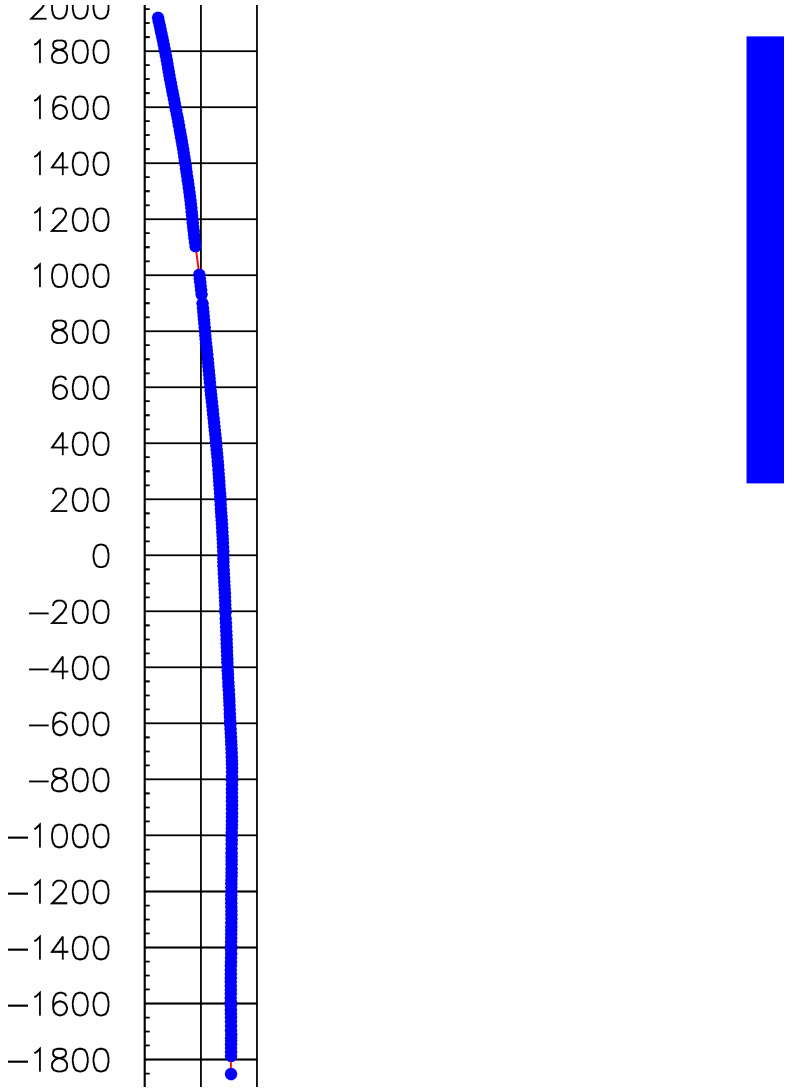


(b) Sample image at $(923.105, -1935.69)$

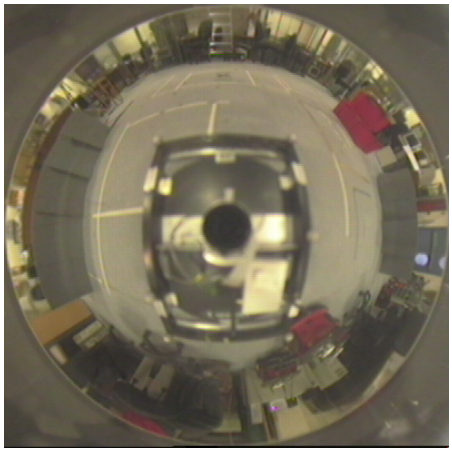


(c) Sample image at $(676.751, 1643.6)$

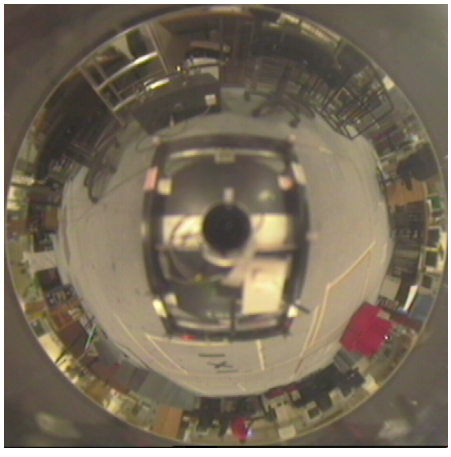
Figure B.19: The manifold STRAIGHT3



(a) Visualisation of parameter space of manifold STRAIGHT4

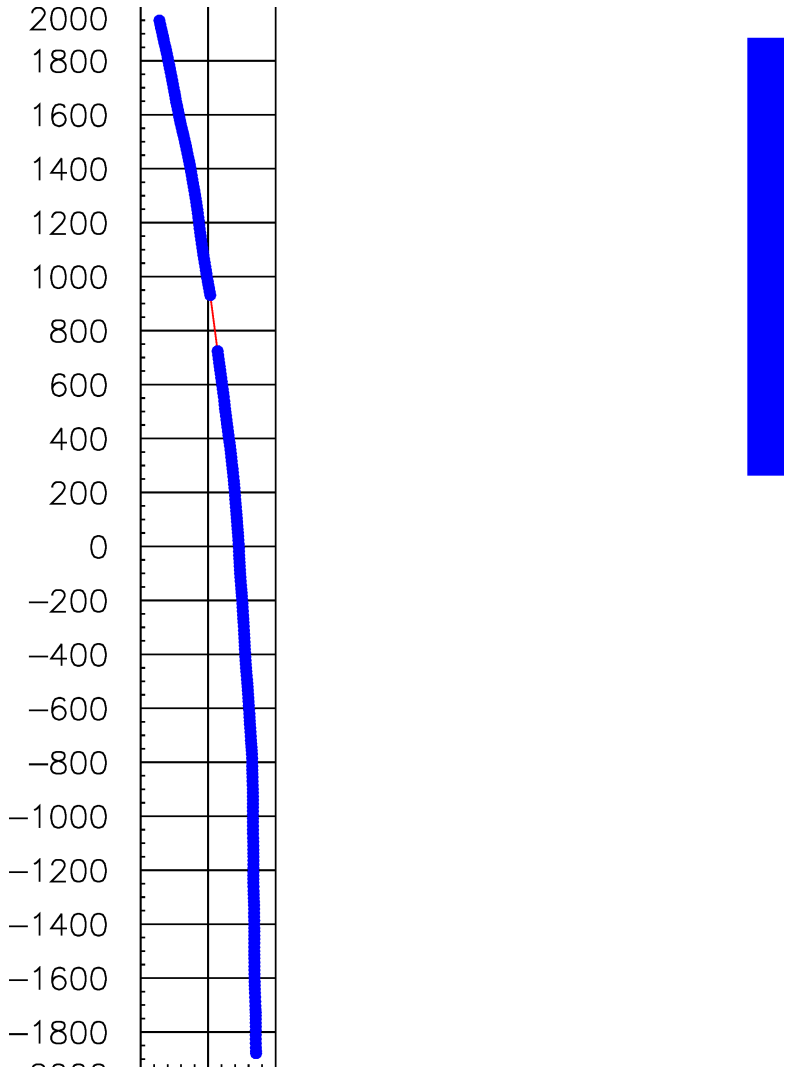


(b) Sample image at $(907.734, -1851.63)$

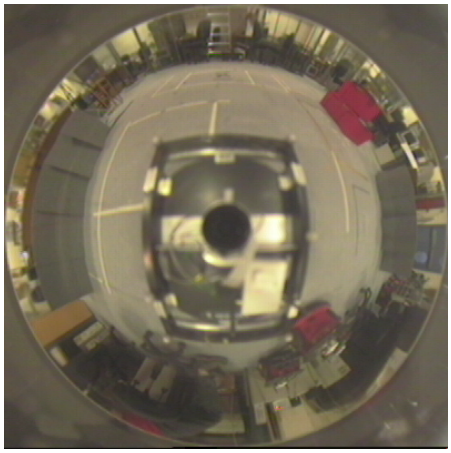


(c) Sample image at $(653.399, 1892)$

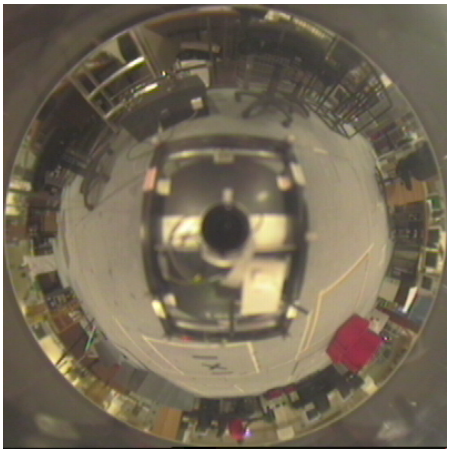
Figure B.20: The manifold STRAIGHT4



(a) Visualisation of parameter space of manifold STRAIGHT5

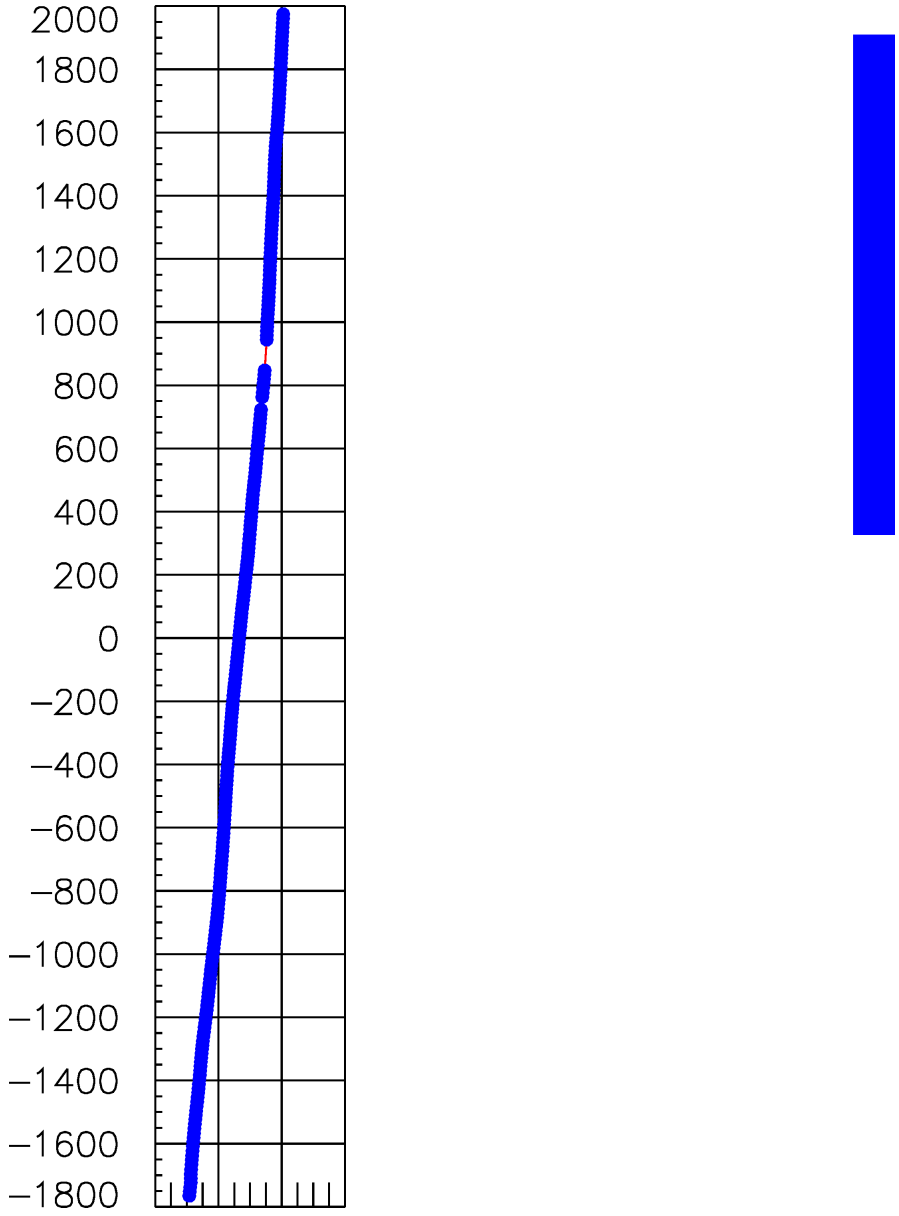


(b) Sample image at $(927.707, -1878.56)$

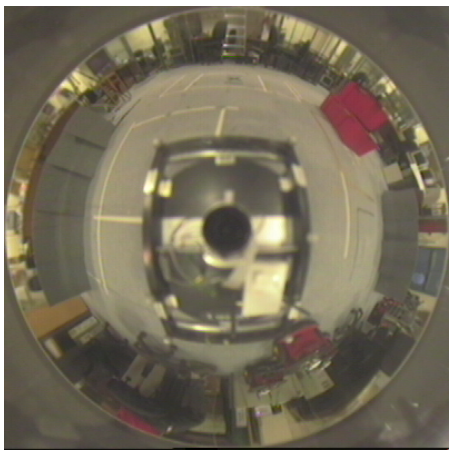


(c) Sample image at $(575.547, 1920.91)$

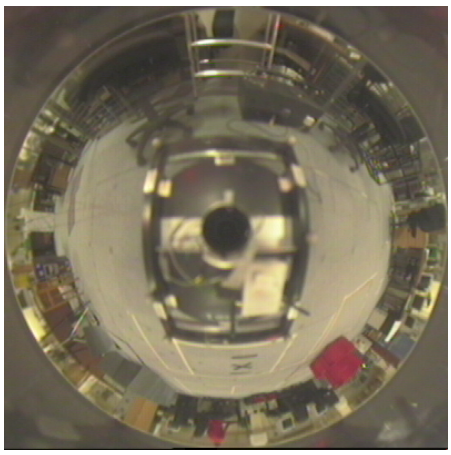
Figure B.21: The manifold STRAIGHT5



(a) Visualisation of parameter space of manifold STRAIGHT6

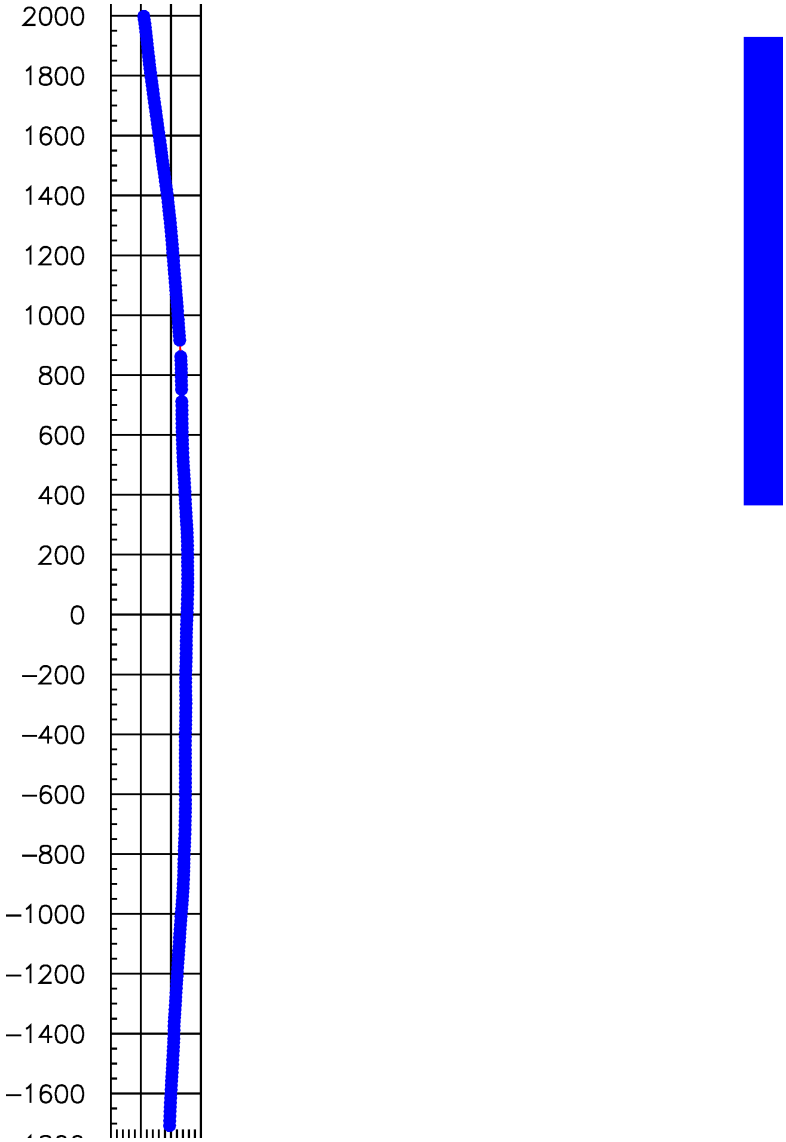


(b) Sample image at $(907.699, -1765.34)$

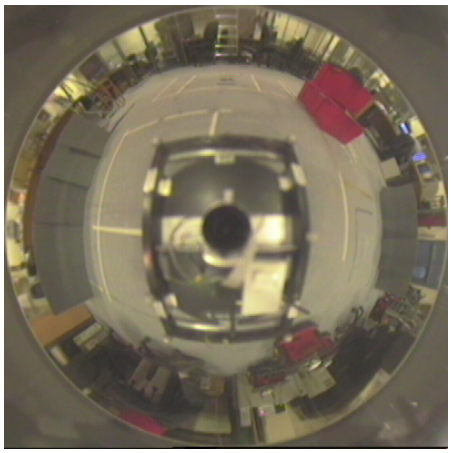


(c) Sample image at $(1203.36, 1946.37)$

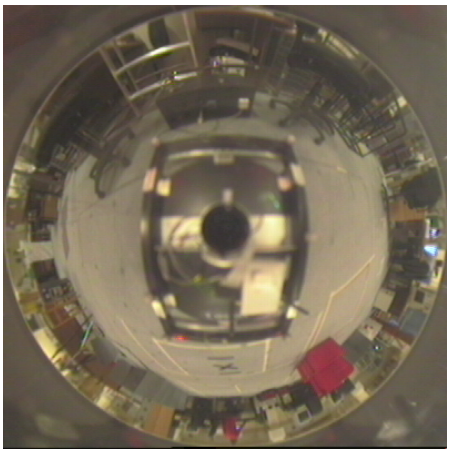
Figure B.22: The manifold STRAIGHT6



(a) Visualisation of parameter space of manifold STRAIGHT7

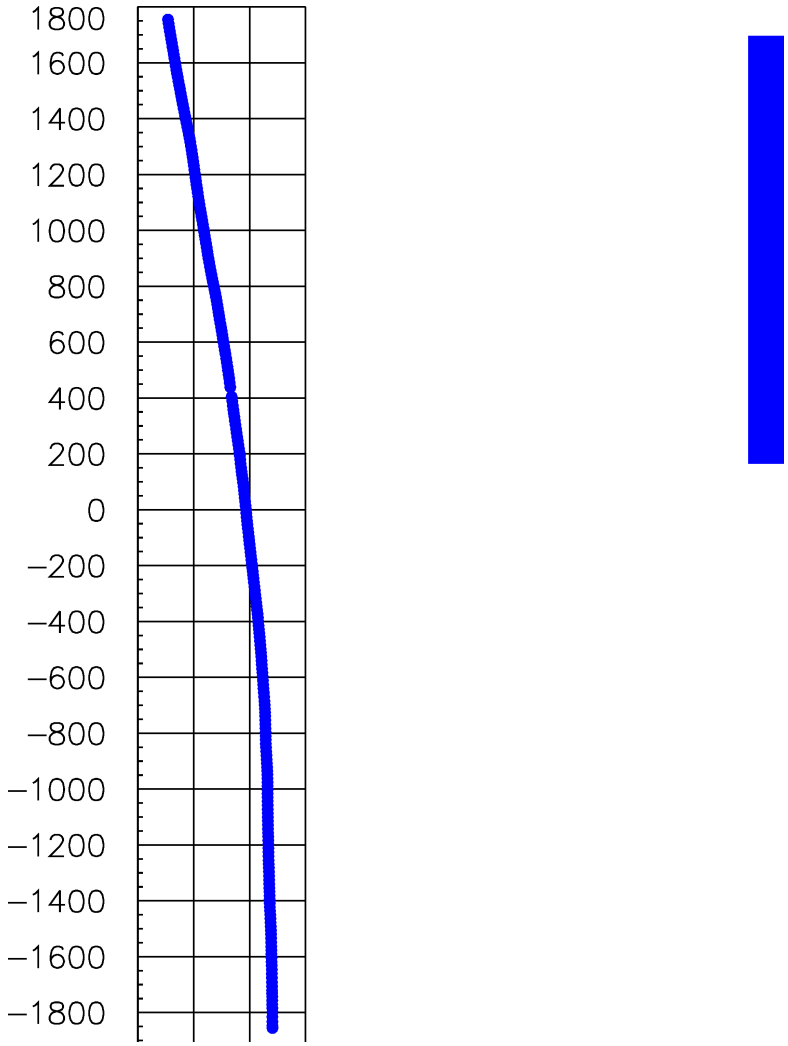


(b) Sample image at $(895.495, -1707.1)$

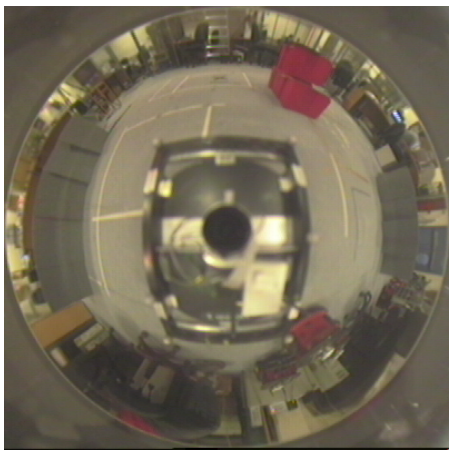


(c) Sample image at $(813.002, 1973.62)$

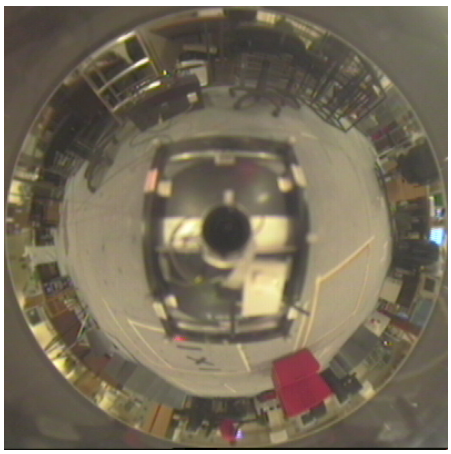
Figure B.23: The manifold STRAIGHT7



(a) Visualisation of parameter space of manifold STRAIGHT8

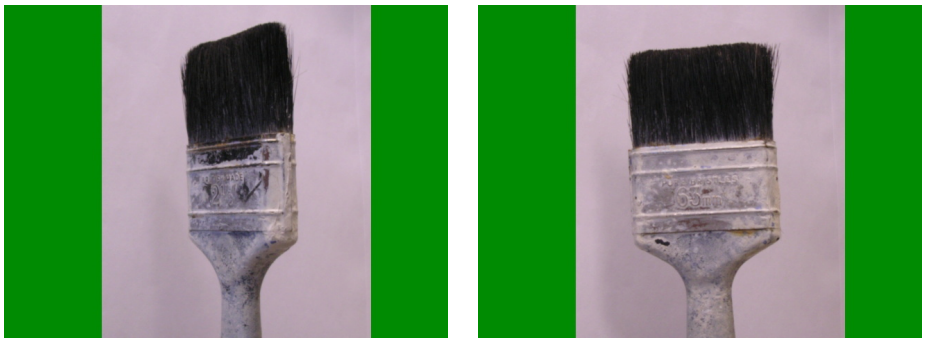


(b) Sample image at $(881.602, -1854.87)$



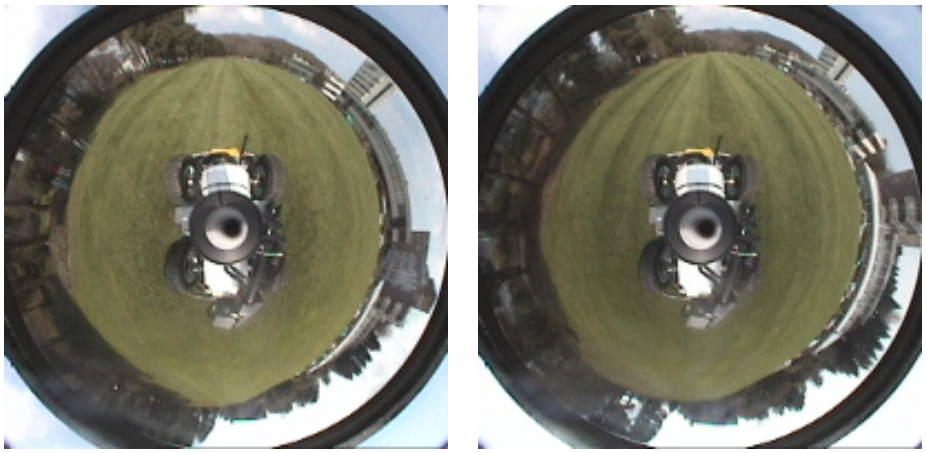
(c) Sample image at $(512.315, 1726.61)$

Figure B.24: The manifold STRAIGHT8



(a) Sample image at (0) (b) Sample image at (136)

Figure B.25: The manifold BRUSH



(a) Sample image at (0) (b) Sample image at (817)

Figure B.26: The manifold IDRISSTRAIGHT

(a) Sample image at (1)

(b) Sample image at (1998)

Figure B.27: The manifold WOODBOX

Appendix C

Metrics for visual navigation

In this appendix we present the results from simulated visual homing on the 1-D omnidirectional camera datasets which were captured from real cameras on real robots. This is intended to replicated existing work (Labrosse, 2006, 2007) and is included as part of the discussion of the metrics we considered in Chapter 5, Section 5.1.5.1.

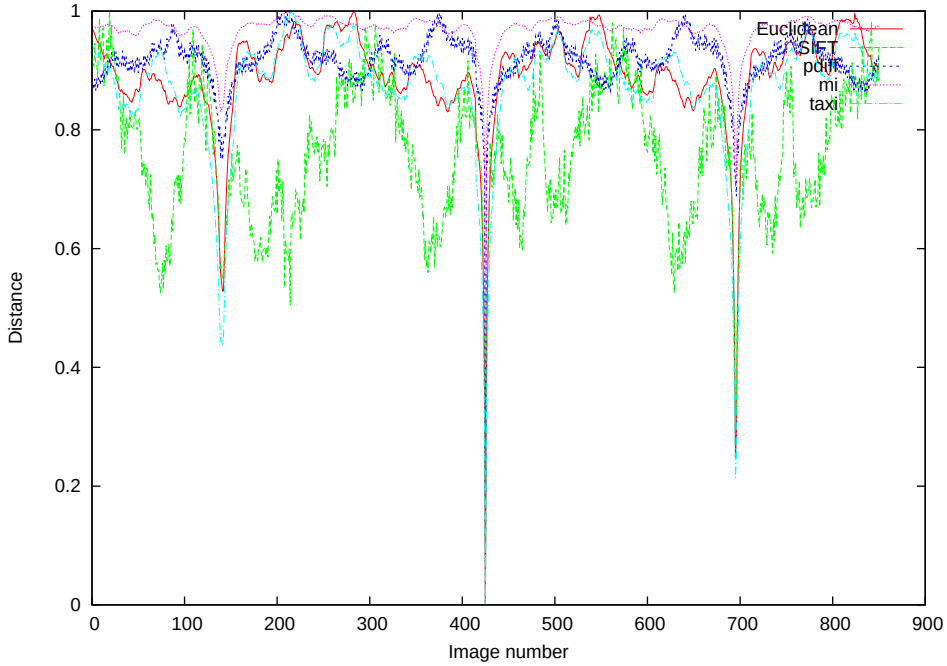


Figure C.1: Distance metrics for visual navigation on the manifold Circle_3

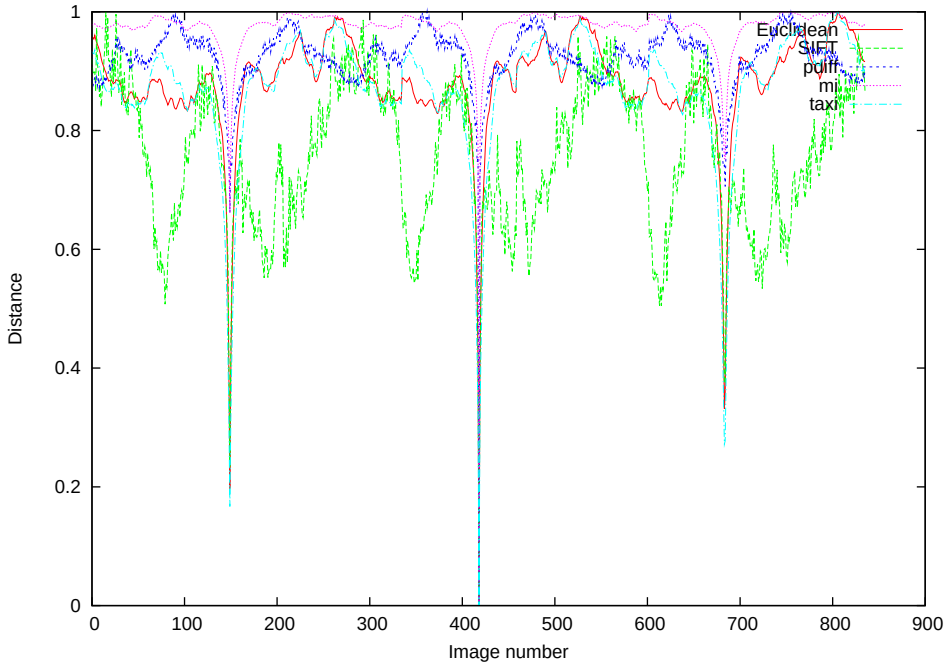


Figure C.2: Distance metrics for visual navigation on the manifold CIRCLE_4

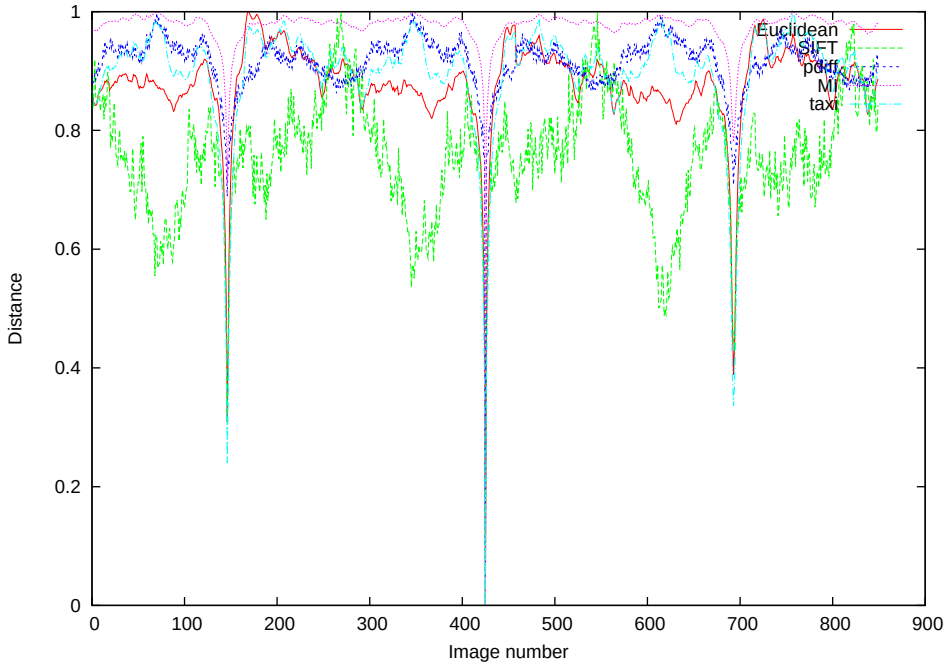


Figure C.3: Distance metrics for visual navigation on the manifold CIRCLE_5

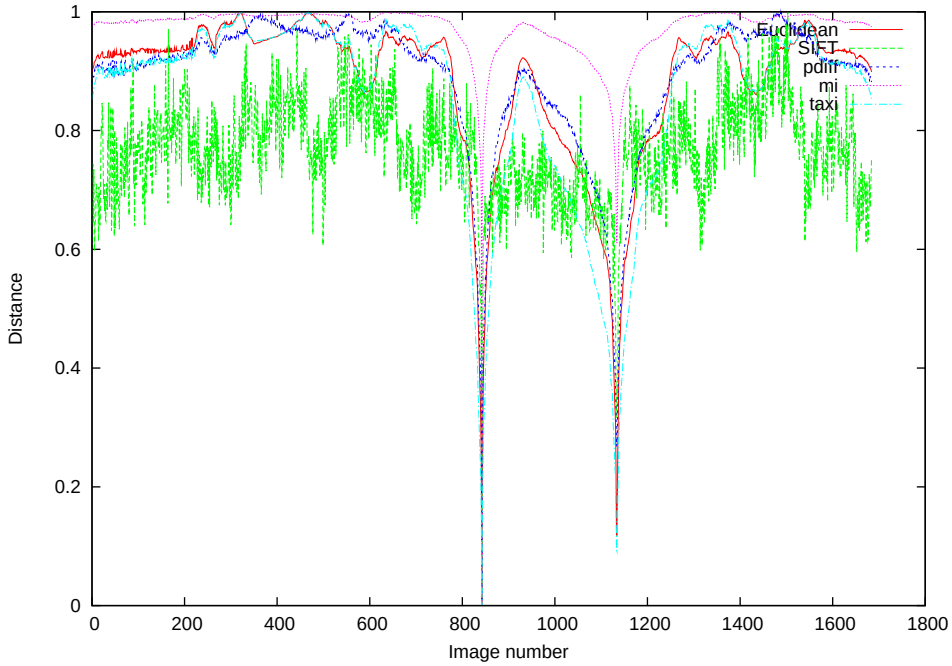


Figure C.4: Distance metrics for visual navigation on the manifold IDRIS_CIRCLE

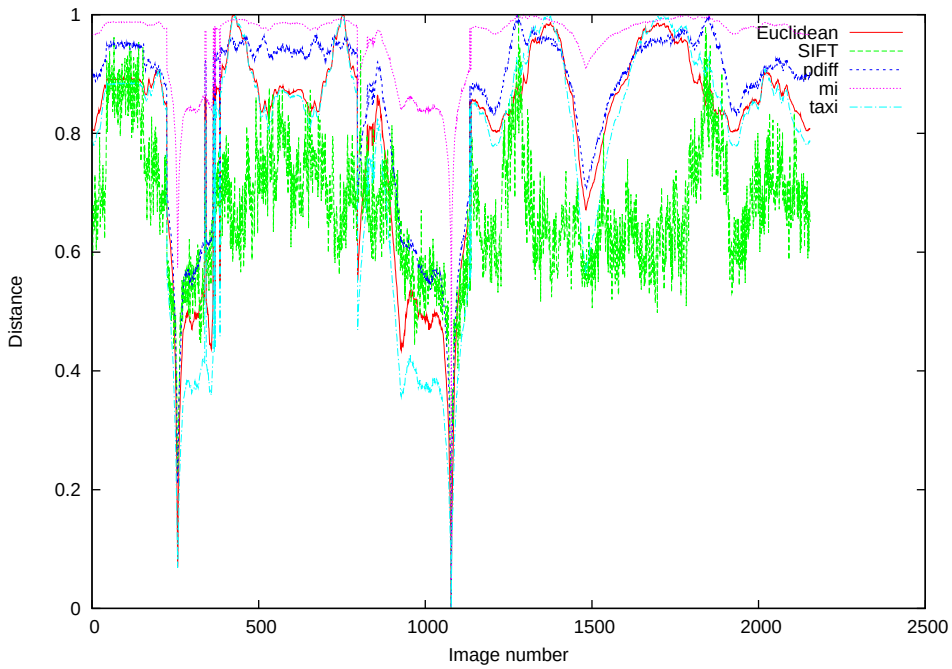


Figure C.5: Distance metrics for visual navigation on the manifold IDRISFIG_8

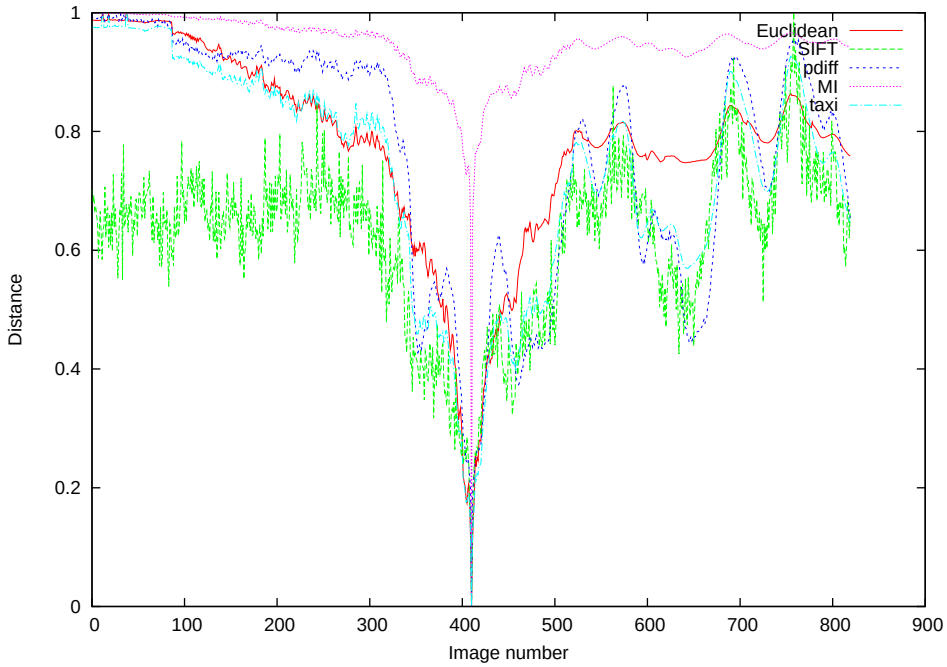


Figure C.6: Distance metrics for visual navigation on the manifold IDRIS_STRAIGHT

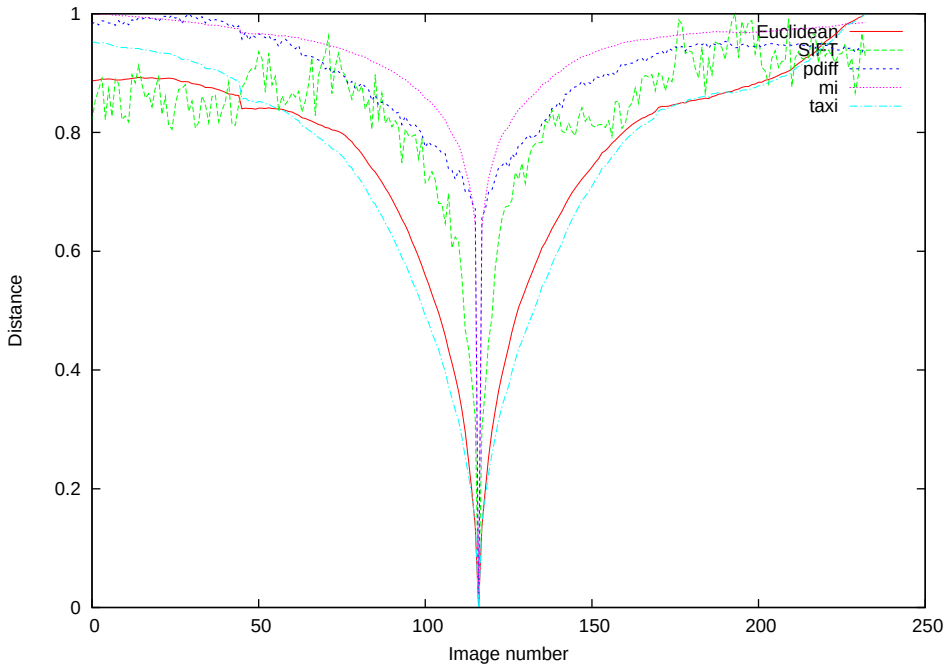


Figure C.7: Distance metrics for visual navigation on the manifold STRAIGHT_1

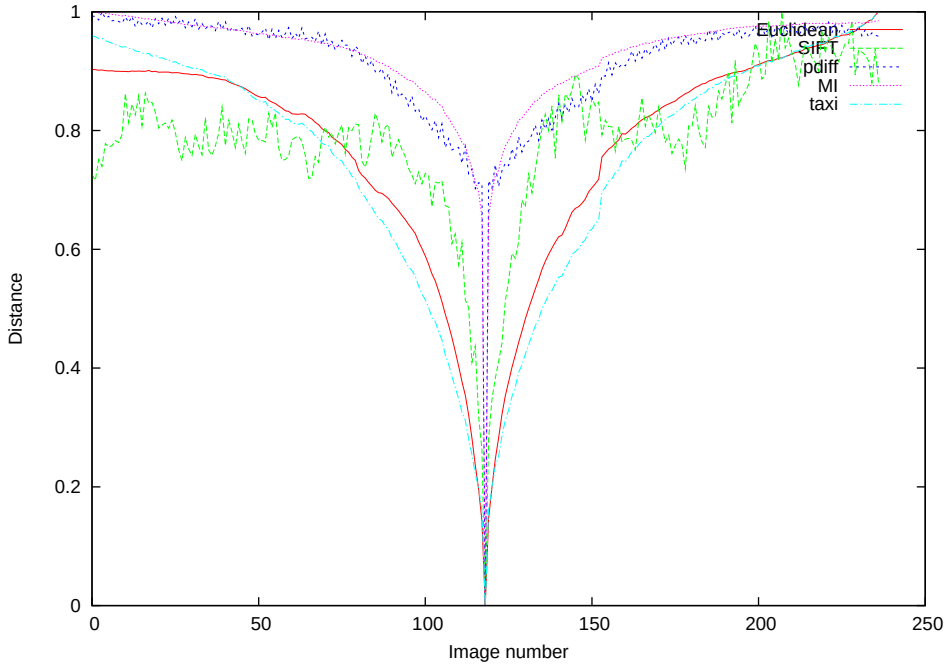


Figure C.8: Distance metrics for visual navigation on the manifold STRAIGHT_2

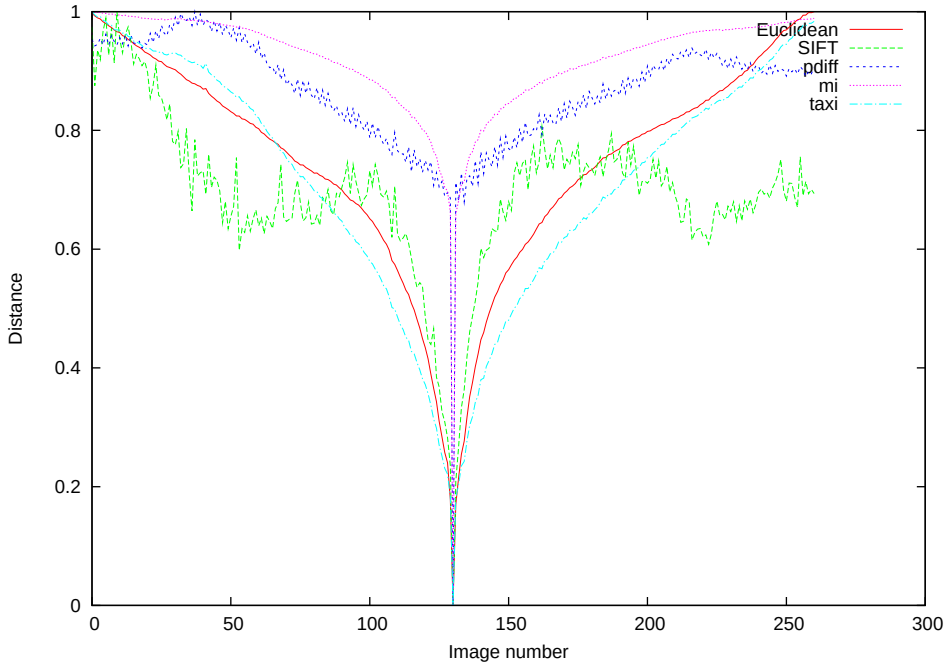


Figure C.9: Distance metrics for visual navigation on the manifold STRAIGHT_3

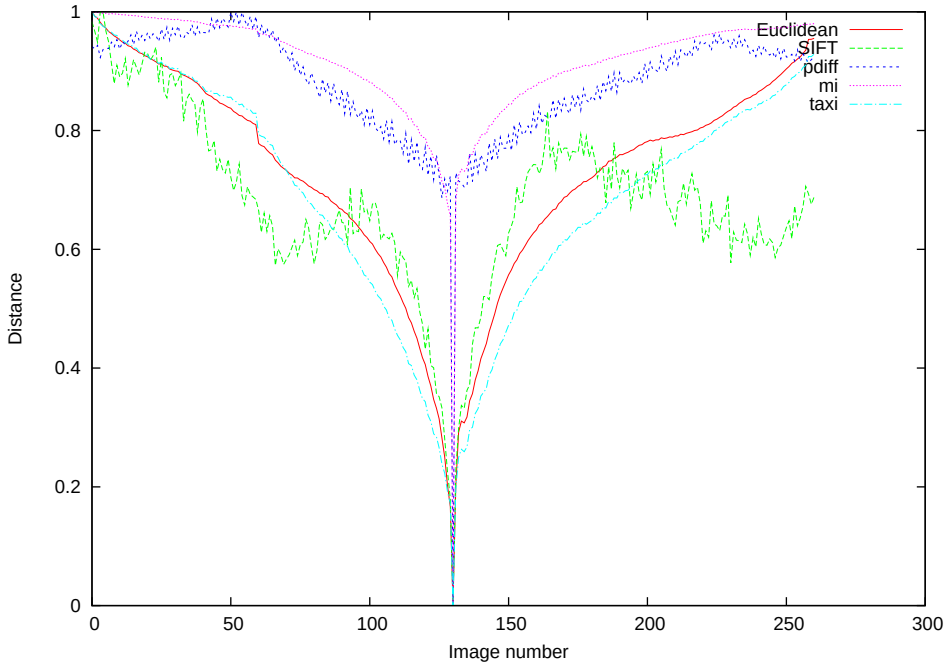


Figure C.10: Distance metrics for visual navigation on the manifold STRAIGHT_4

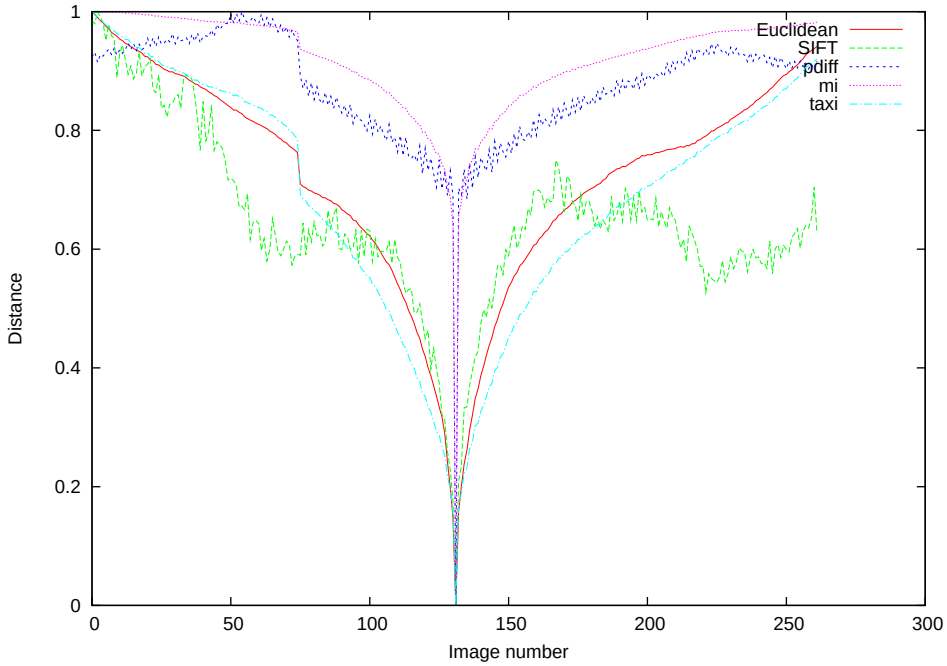


Figure C.11: Distance metrics for visual navigation on the manifold STRAIGHT_5

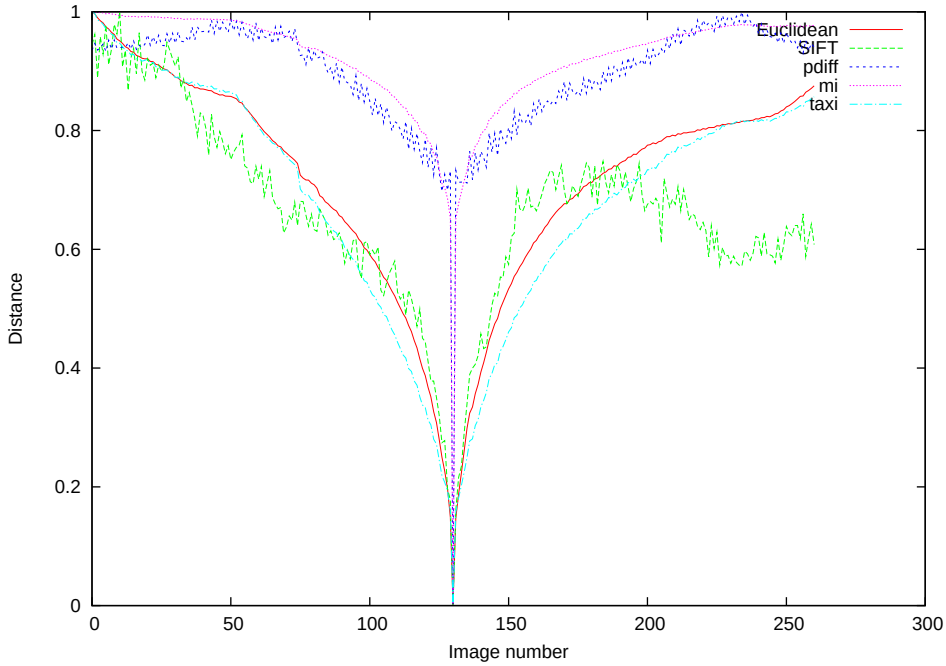


Figure C.12: Distance metrics for visual navigation on the manifold STRAIGHT_6

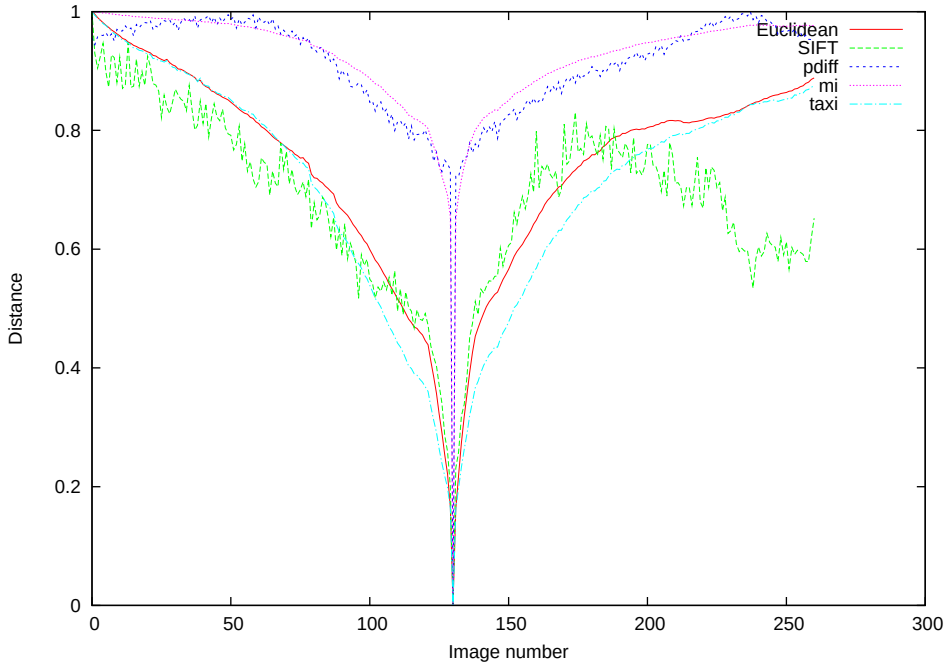


Figure C.13: Distance metrics for visual navigation on the manifold STRAIGHT_7

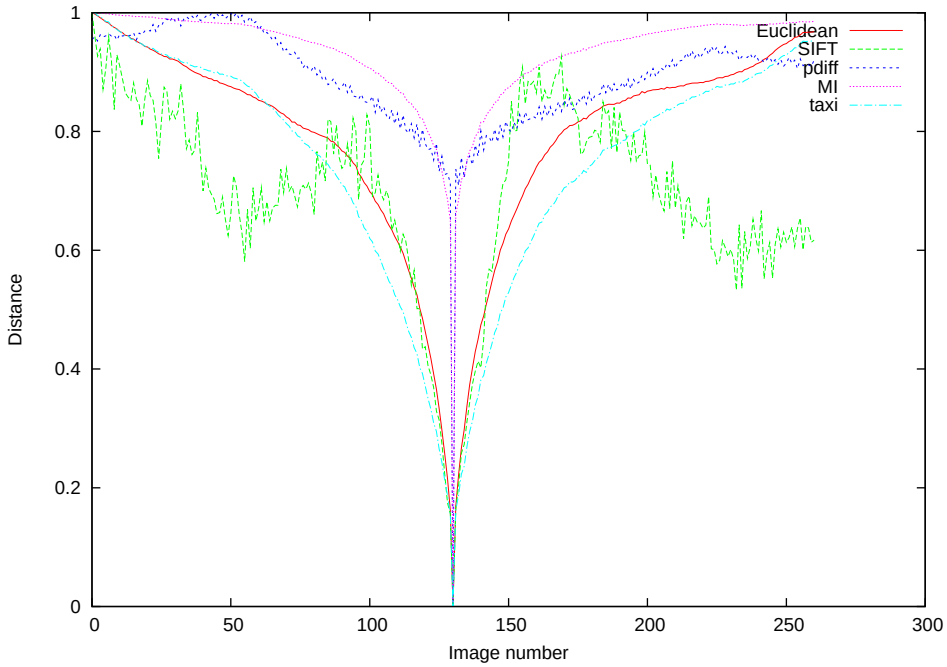
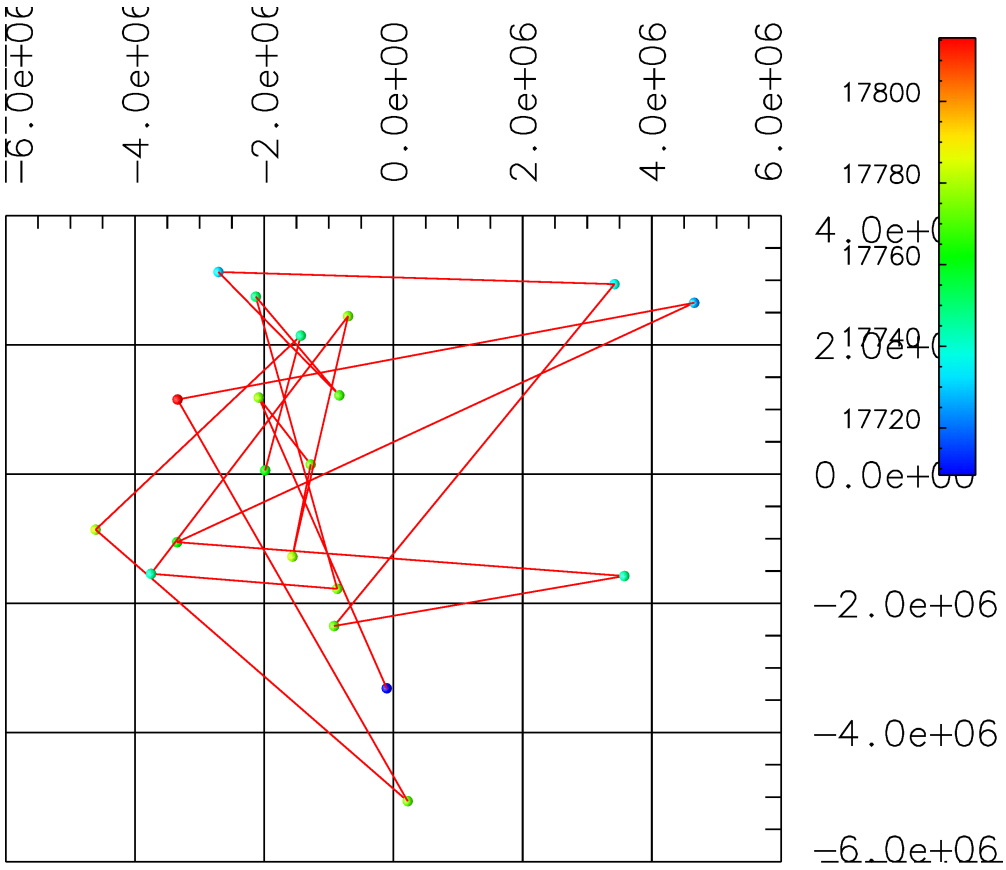


Figure C.14: Distance metrics for visual navigation on the manifold STRAIGHT_8

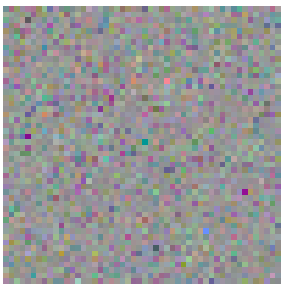
Appendix D

PCA plots of manifolds

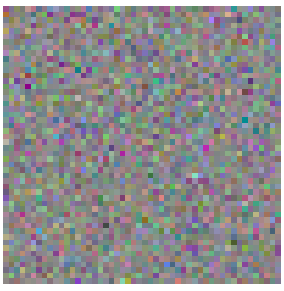
Here we introduce the PCA plots from all of the datasets we are using throughout the thesis. This is based upon the discussion in Chapter 7. Results are identified as to which of the two methods was used, using the same terminology as in Chapter 7.



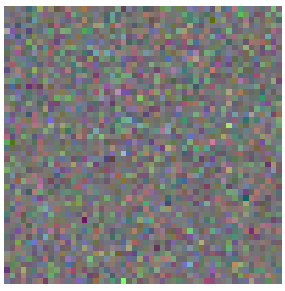
(a) PCA Plot



(b) First principal component

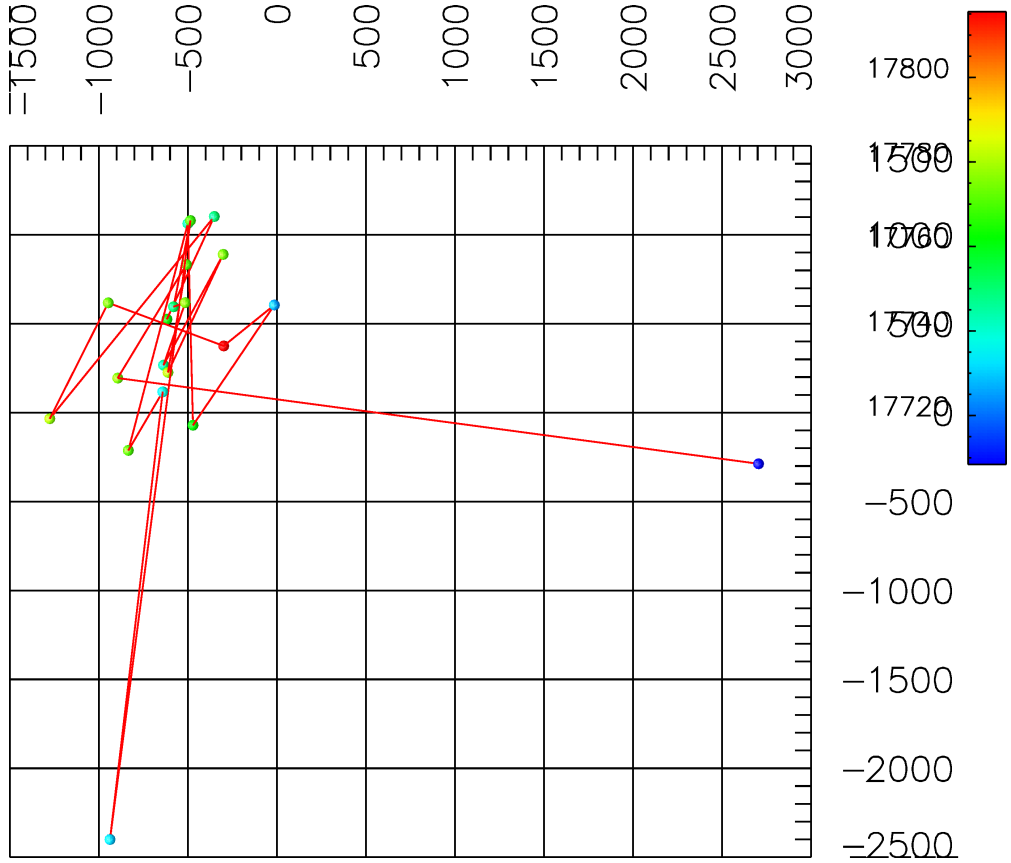


(c) Second principal component

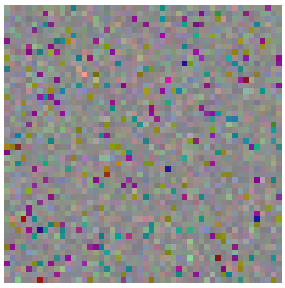


(d) Third principal component

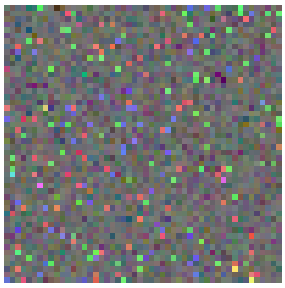
Figure D.1: The manifold Bmnoise, 'mtfast'



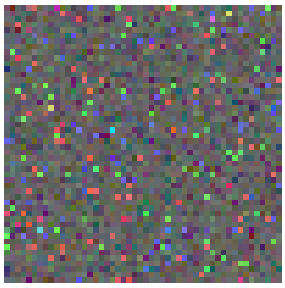
(a) PCA Plot



(b) First principal component

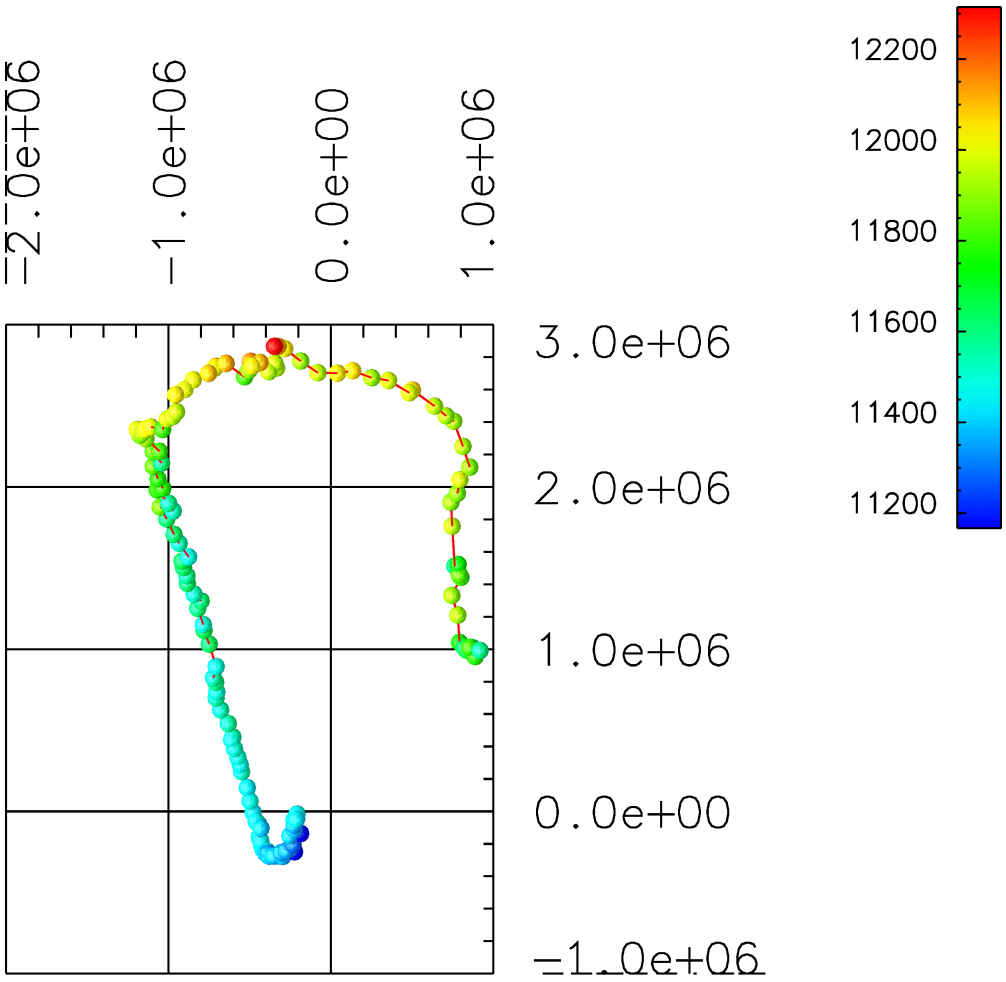


(c) Second principal component

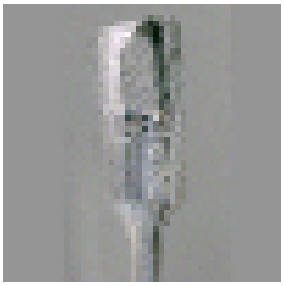


(d) Third principal component

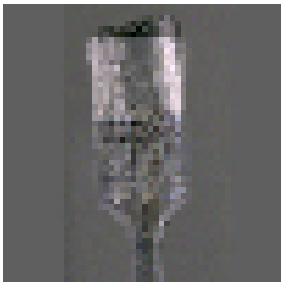
Figure D.2: The manifold Bmnoise, 'impca'



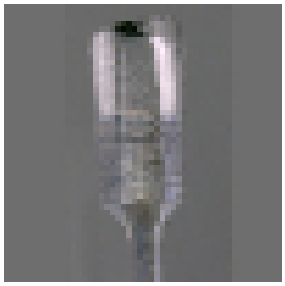
(a) PCA Plot



(b) First principal component

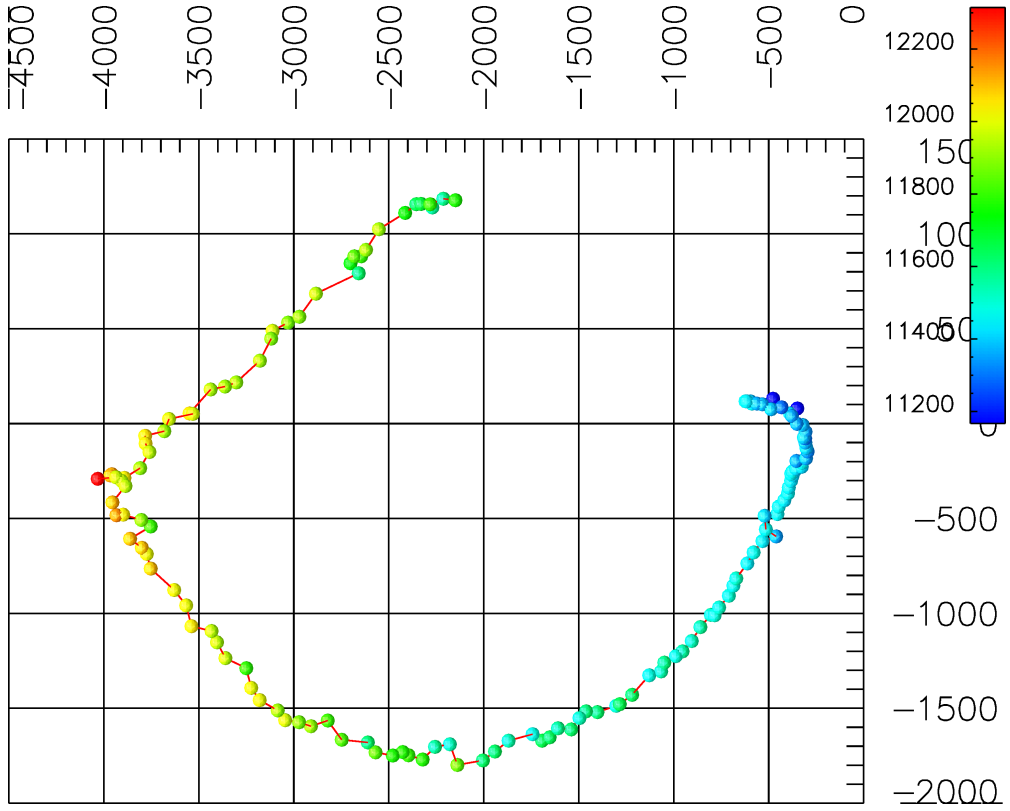


(c) Second principal component

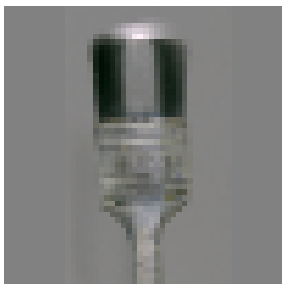


(d) Third principal component

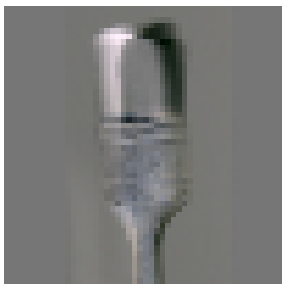
Figure D.3: The manifold Brush, 'mtfast'



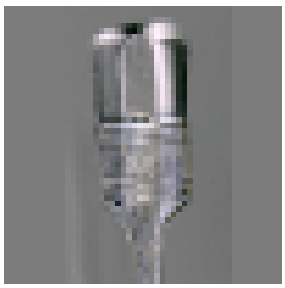
(a) PCA Plot



(b) First principal component

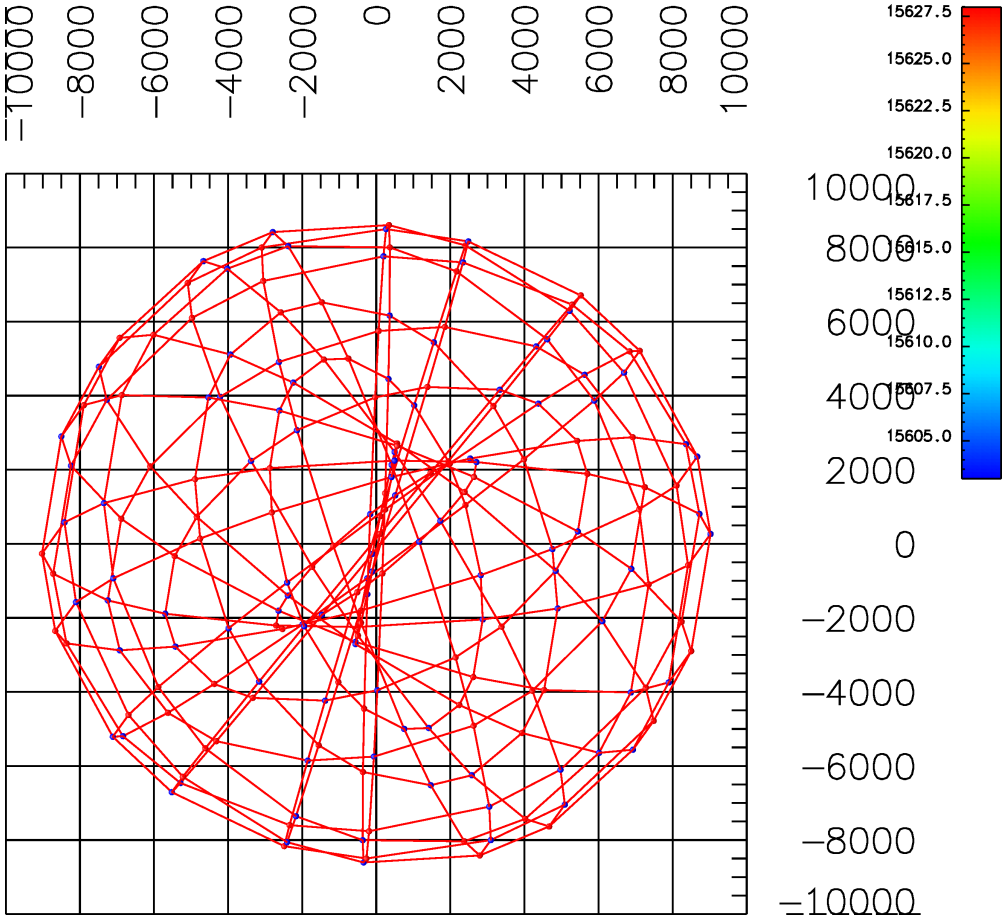


(c) Second principal component

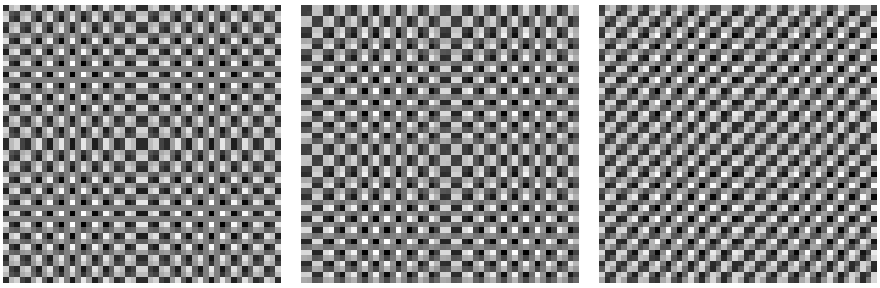


(d) Third principal component

Figure D.4: The manifold Brush, 'impca'



(a) PCA Plot

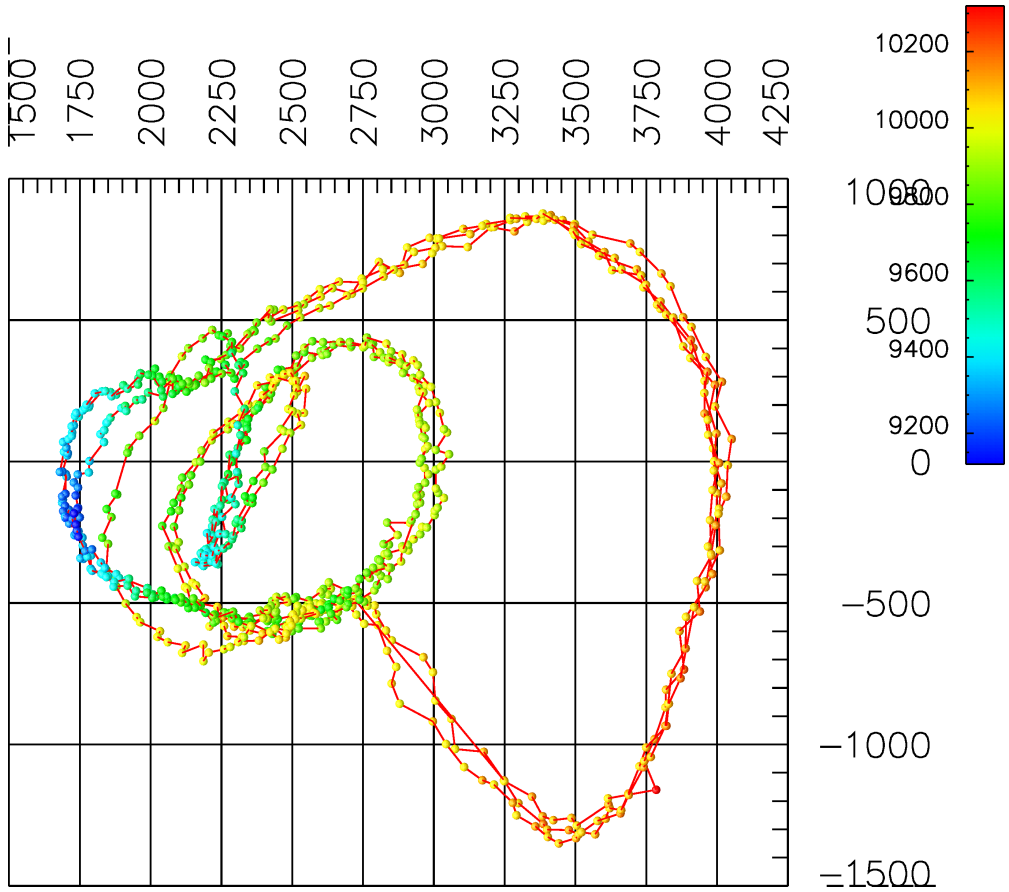


(b) First principal component

(c) Second principal component

(d) Third principal component

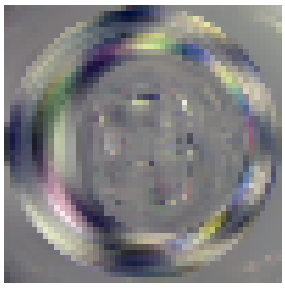
Figure D.5: The manifold Chessboard, 'impca'



(a) PCA Plot

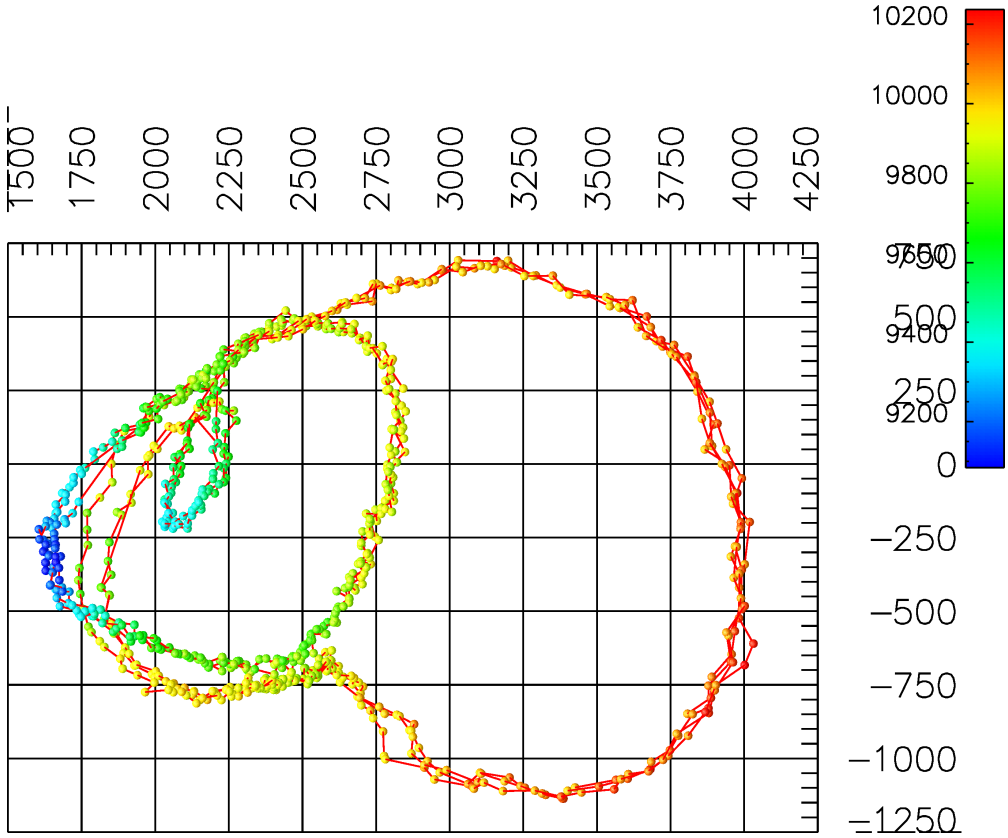
(b) First principal component

(c) Second principal component



(d) Third principal component

Figure D.6: The manifold Circle_3, 'impca'



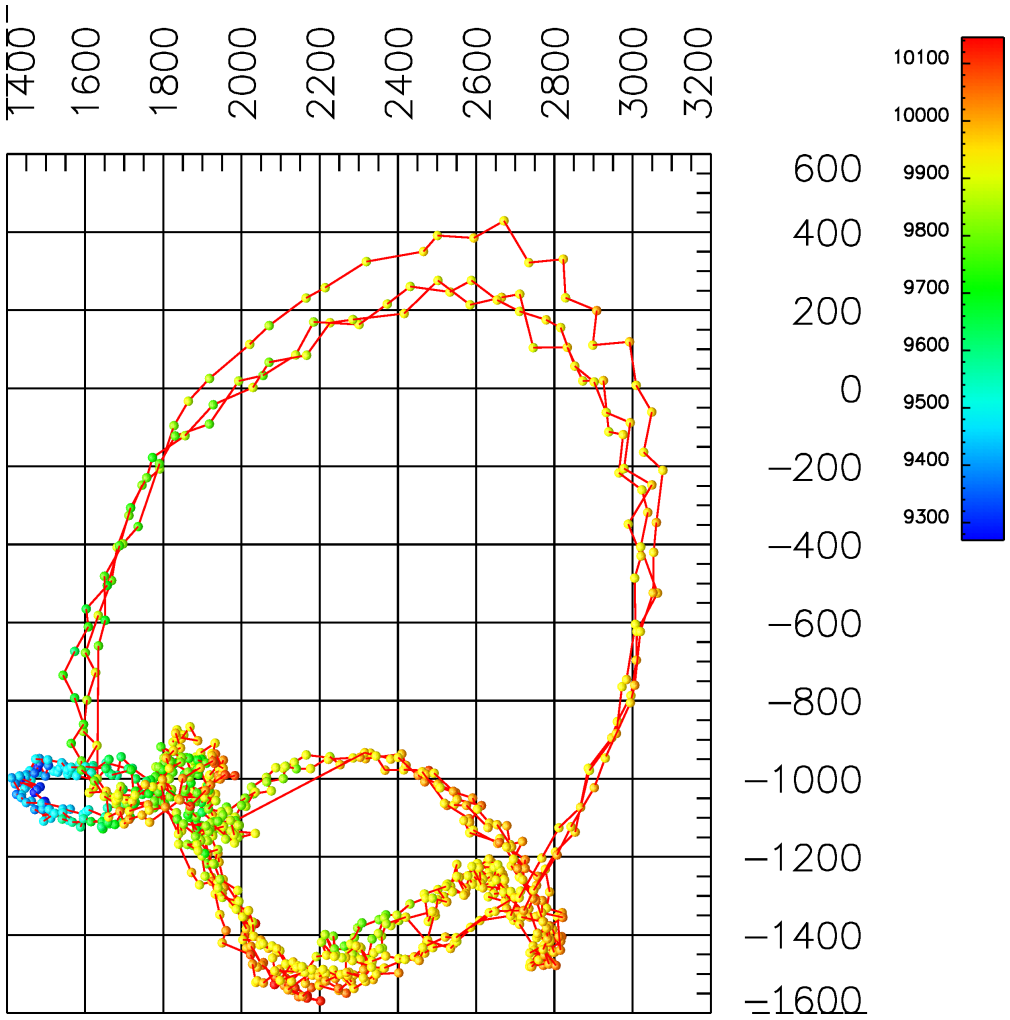
(a) PCA Plot

(b) First principal component

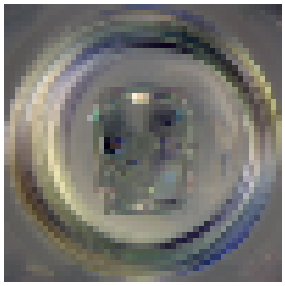
(c) Second principal component

(d) Third principal component

Figure D.7: The manifold Circle_4, 'impca'



(a) PCA Plot

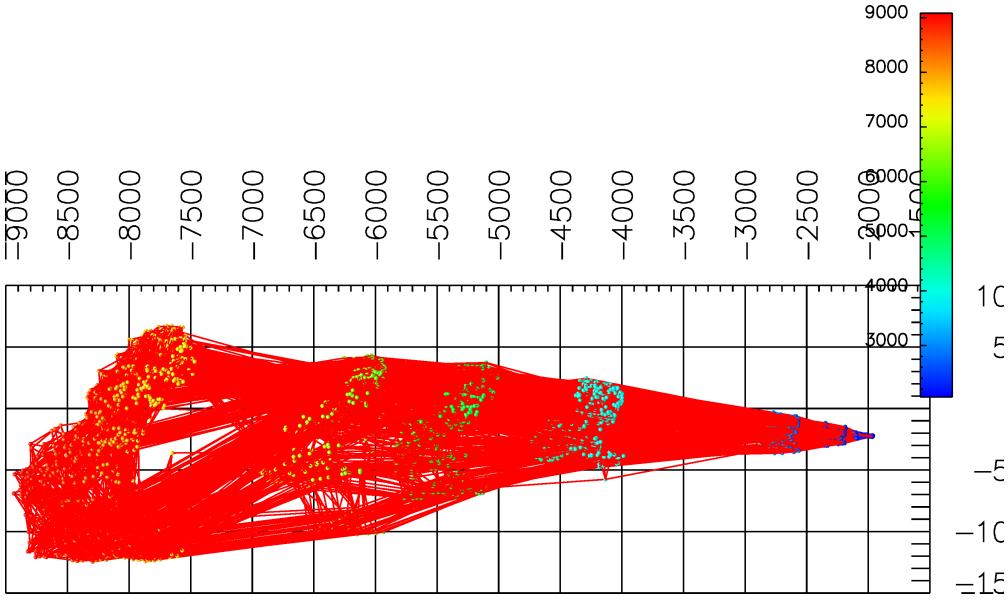


(b) First principal component

(c) Second principal component

(d) Third principal component

Figure D.8: The manifold Circle_5, 'impca'



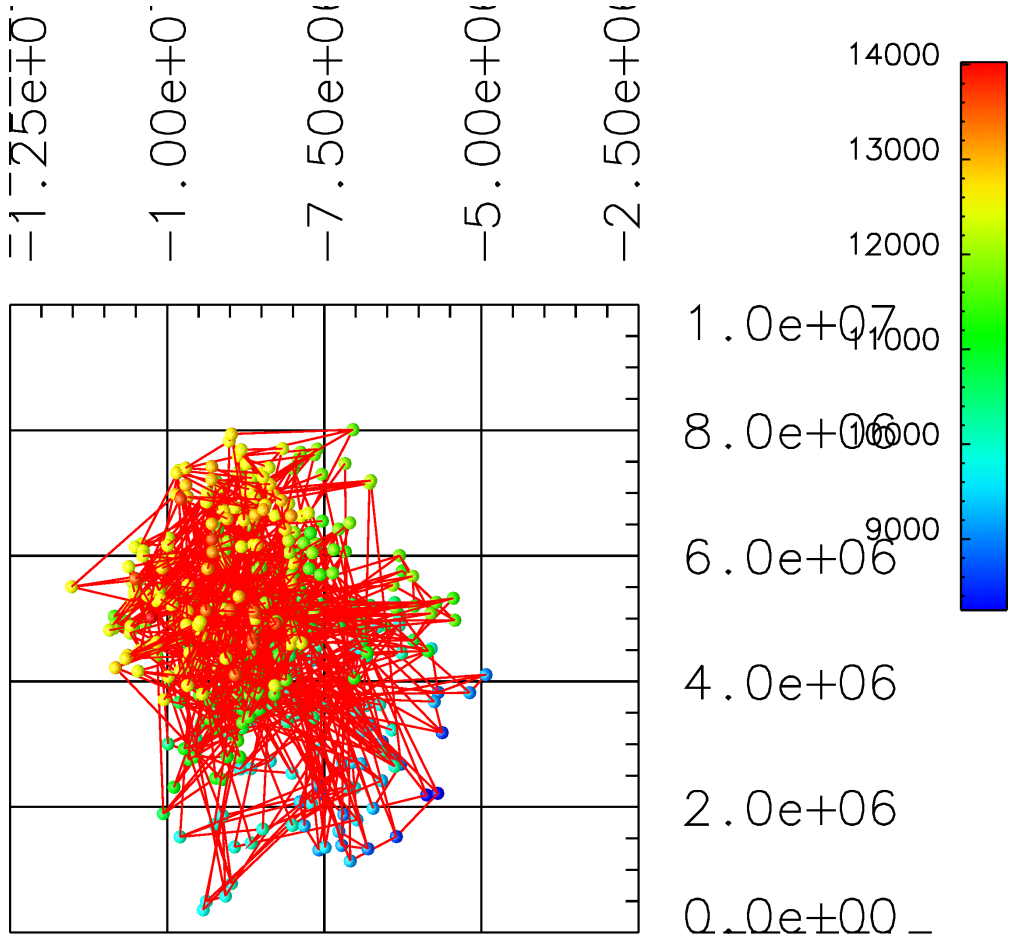
(a) PCA Plot

(b) First principal component

(c) Second principal component

(d) Third principal component

Figure D.9: The manifold Expt03, 'impca'



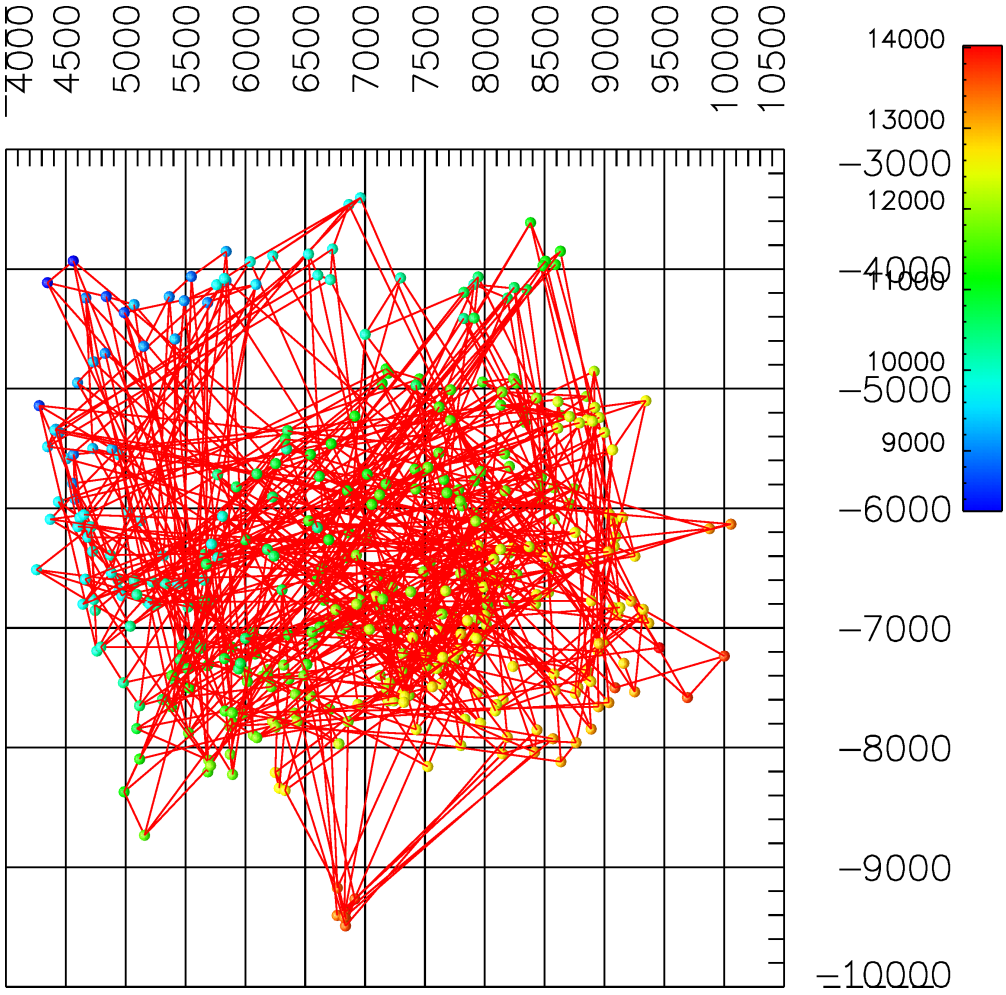
(a) PCA Plot

(b) First principal component

(c) Second principal component

(d) Third principal component

Figure D.10: The manifold Faces, 'mtfast'



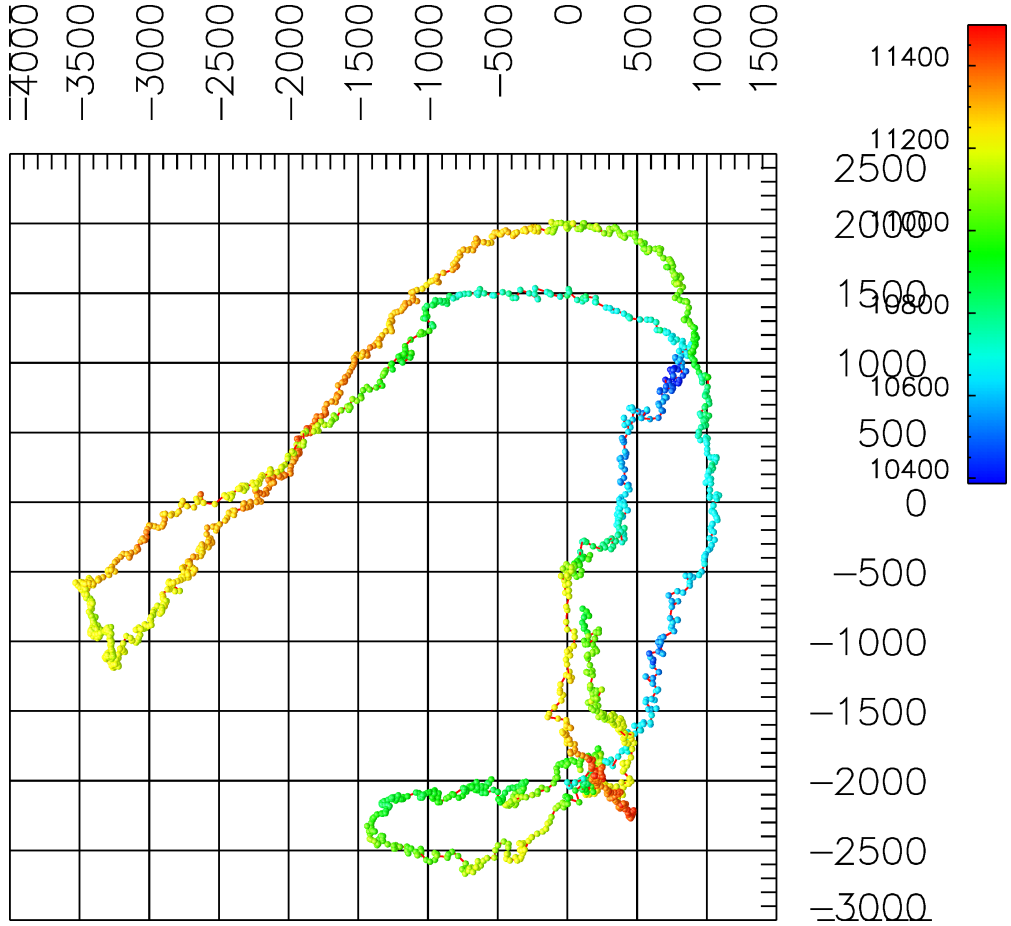
(a) PCA Plot

(b) First principal component

(c) Second principal component

(d) Third principal component

Figure D.11: The manifold FACES, 'impca'



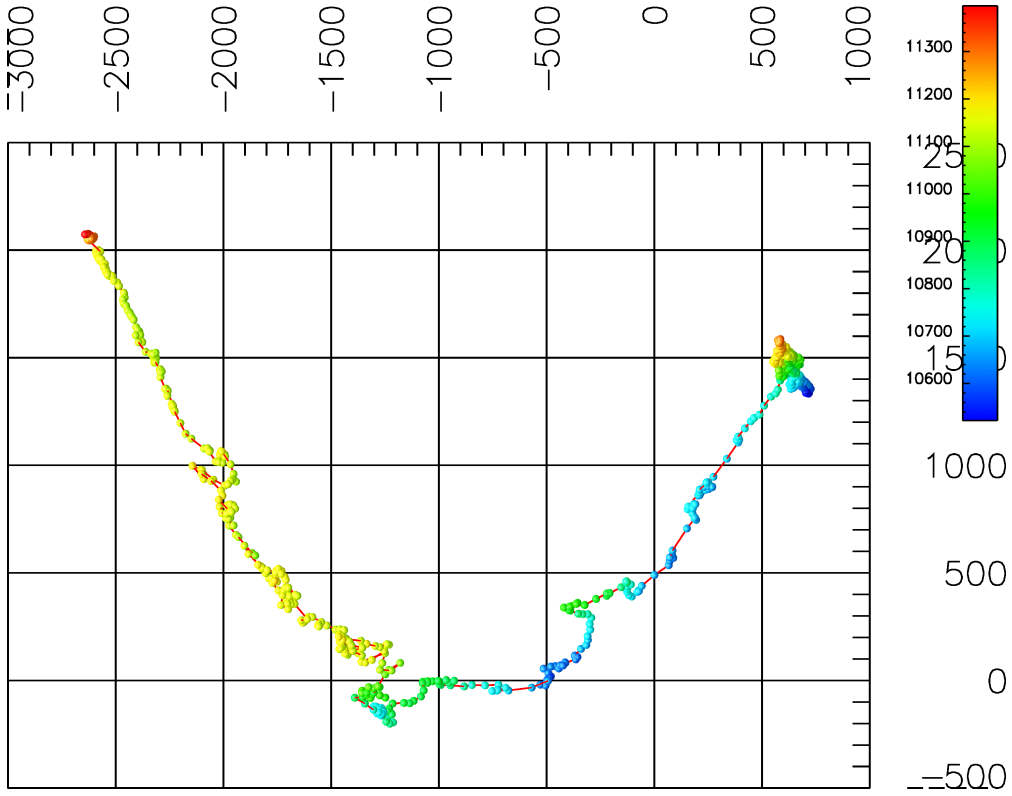
(a) PCA Plot

(b) First principal component

(c) Second principal component

(d) Third principal component

Figure D.12: The manifold IDRISFIG_8, 'impca'



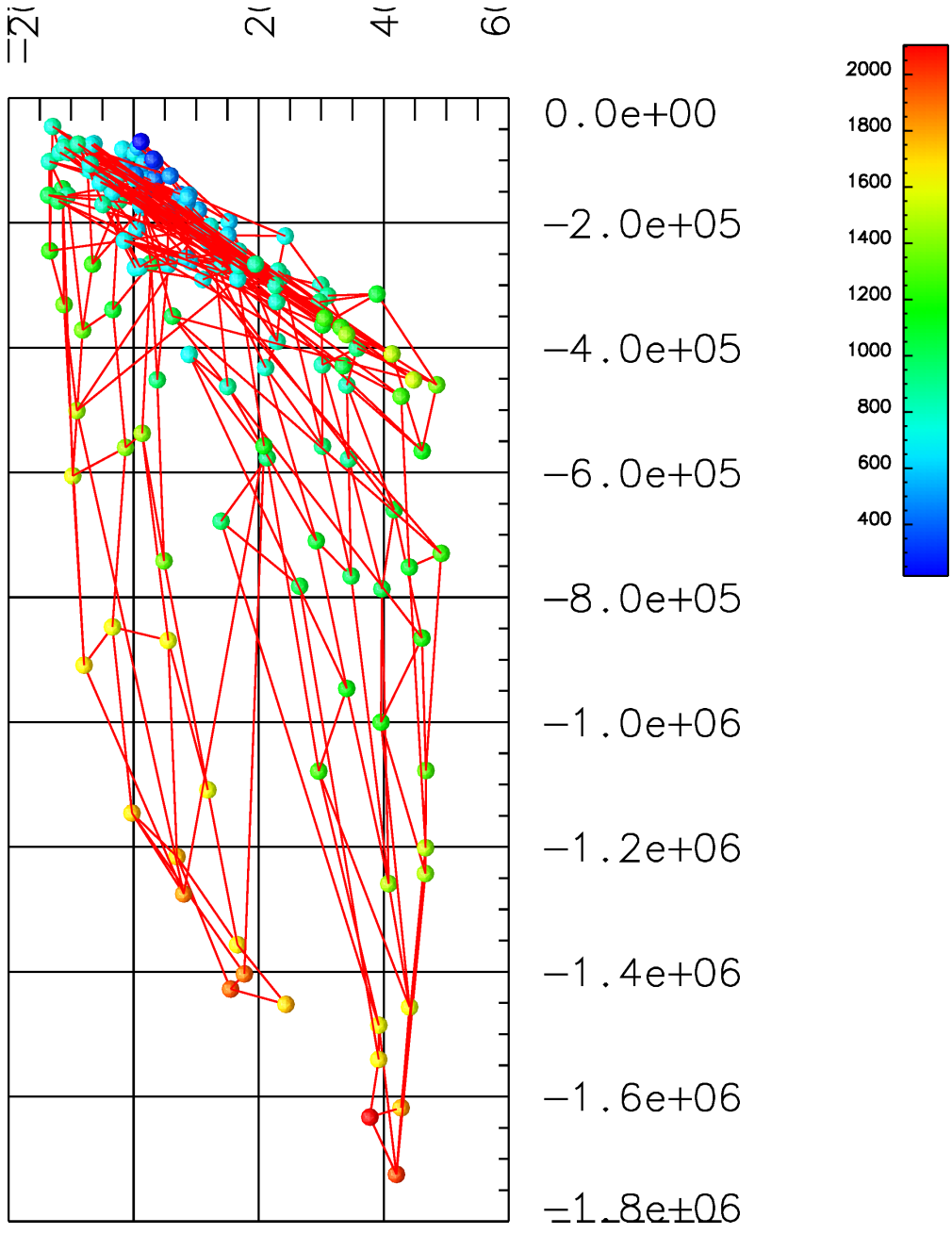
(a) PCA Plot

(b) First principal component

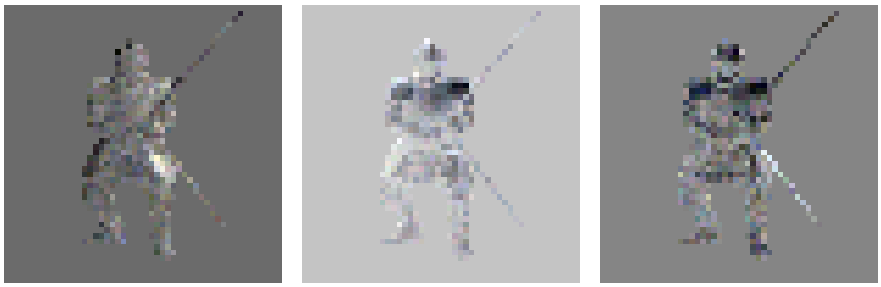
(c) Second principal component

(d) Third principal component

Figure D.13: The manifold IDRIS_STRAIGHT, 'impca'



(a) PCA Plot

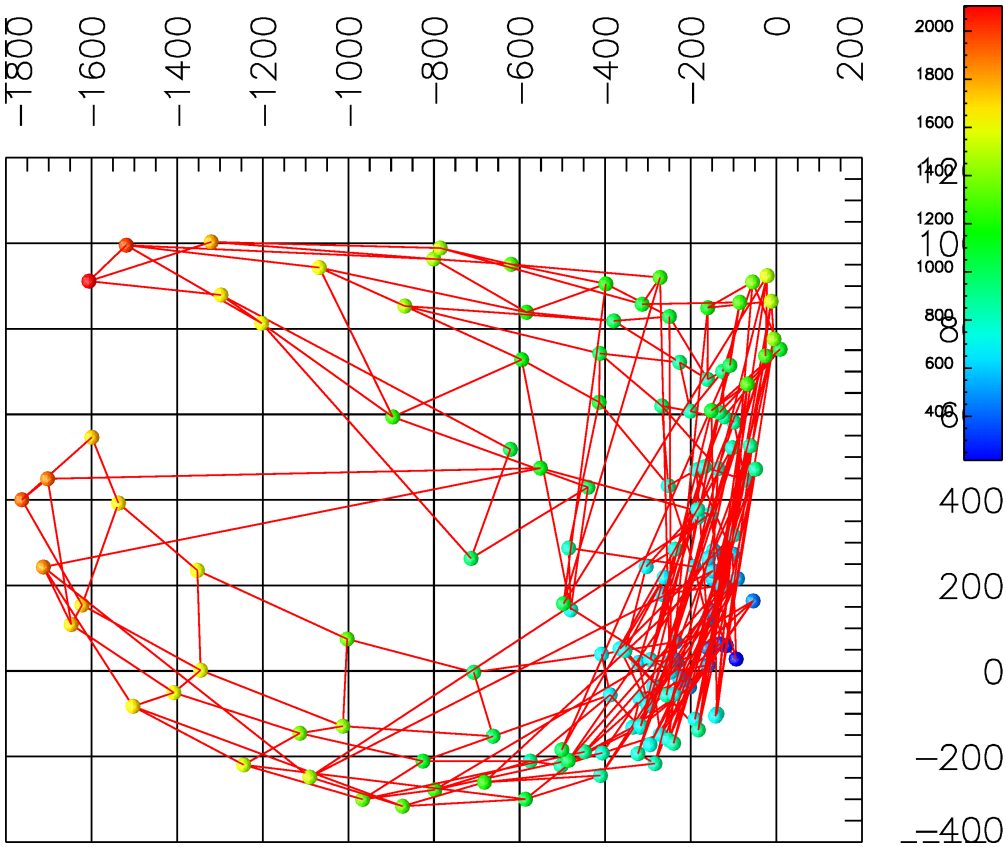


(b) First principal component

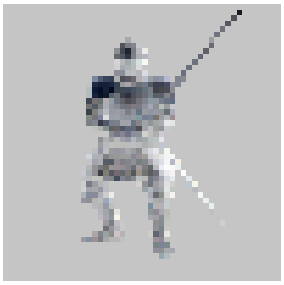
(c) Second principal component

(d) Third principal component

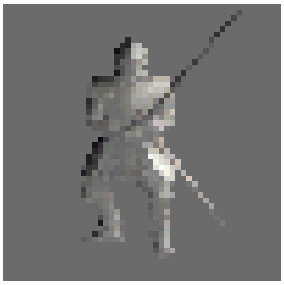
Figure D.14: The manifold KNIGHT_FIGHTING, 'mtfast'



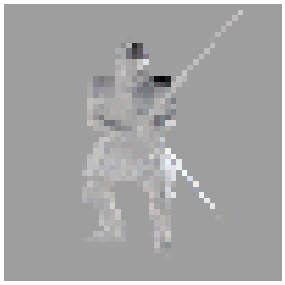
(a) PCA Plot



(b) First principal component

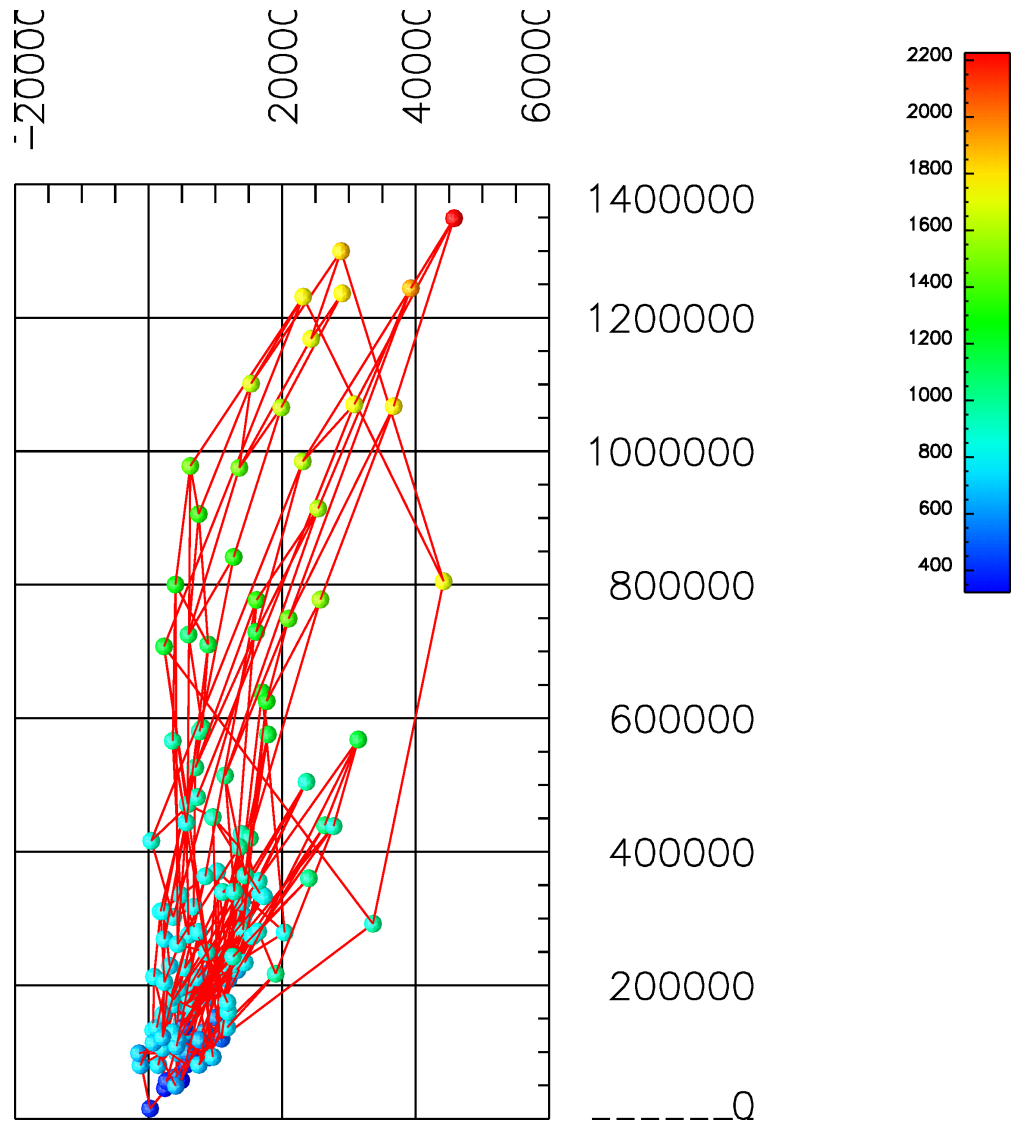


(c) Second principal component

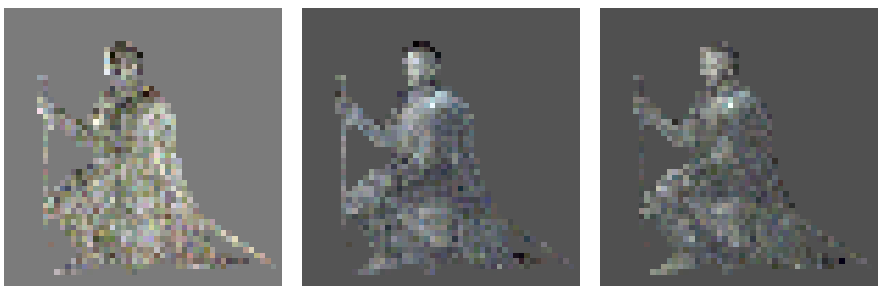


(d) Third principal component

Figure D.15: The manifold KNIGHT_FIGHTING, 'impca'



(a) PCA Plot

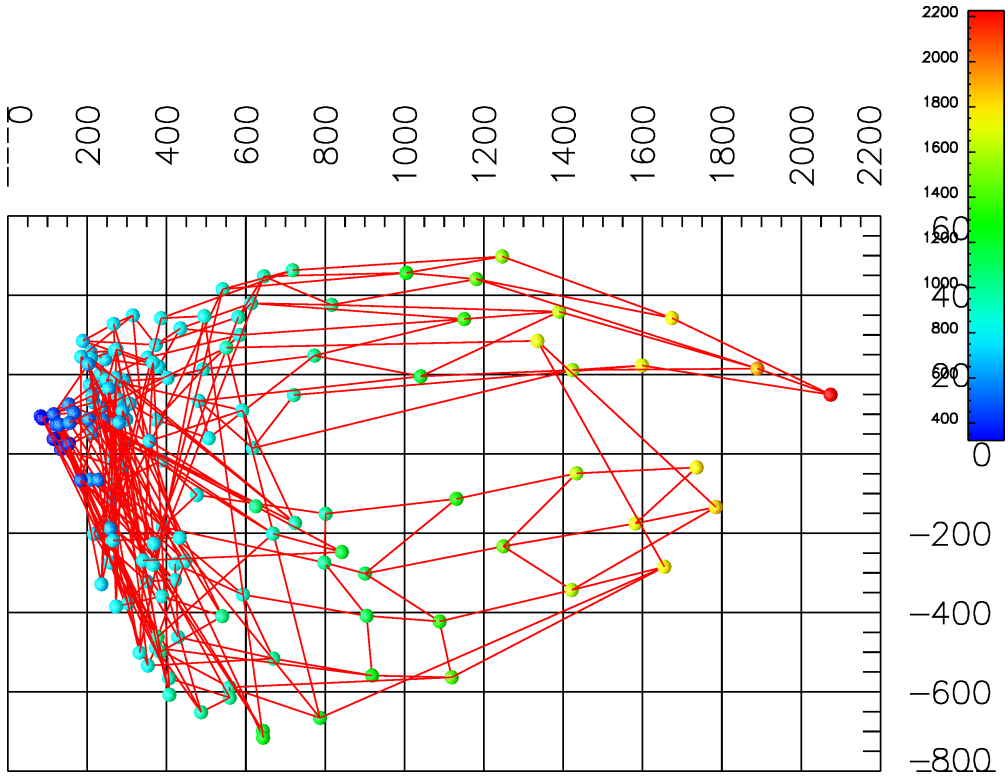


(b) First principal component

(c) Second principal component

(d) Third principal component

Figure D.16: The manifold KNIGHT_KNEELING, 'mtfast'



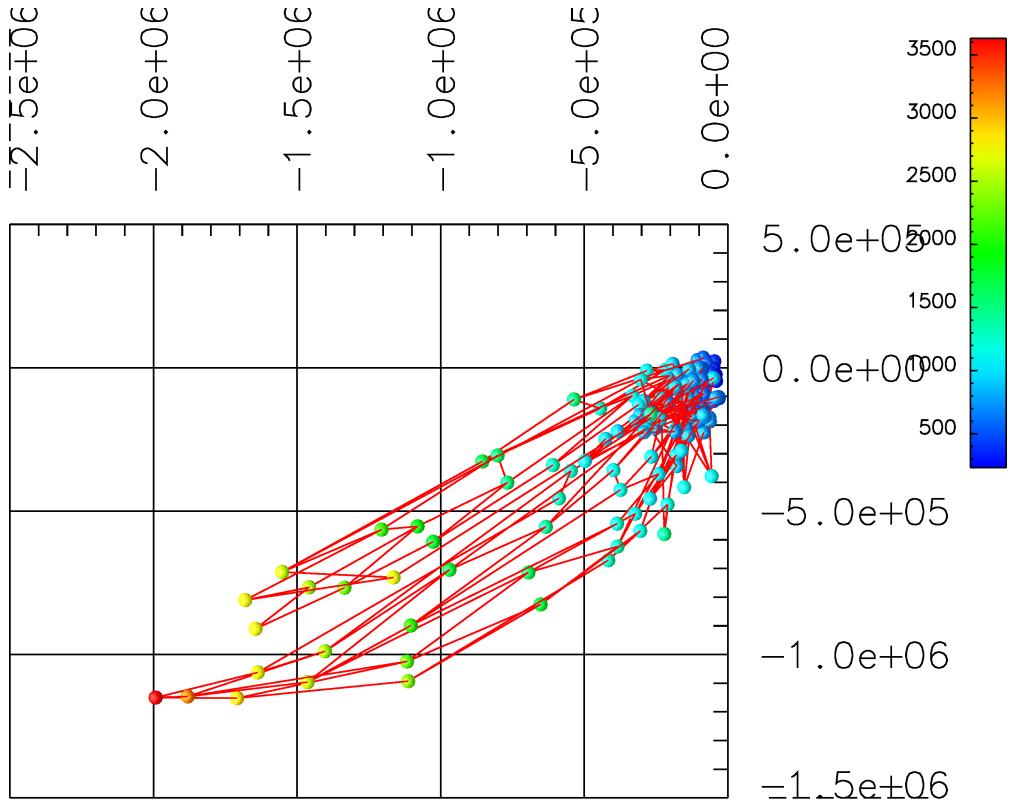
(a) PCA Plot

(b) First principal component

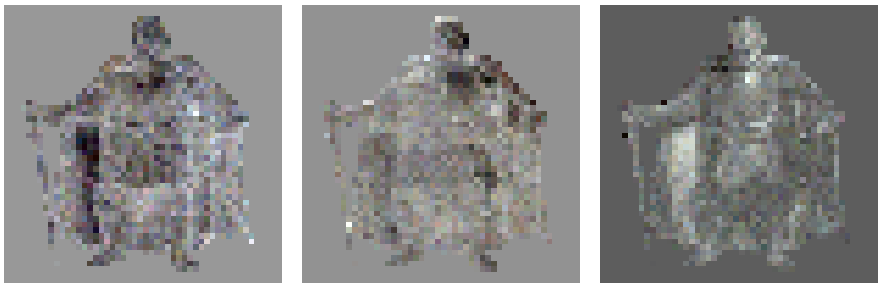
(c) Second principal component

(d) Third principal component

Figure D.17: The manifold KNIGHT_KNEELING, 'impca'



(a) PCA Plot

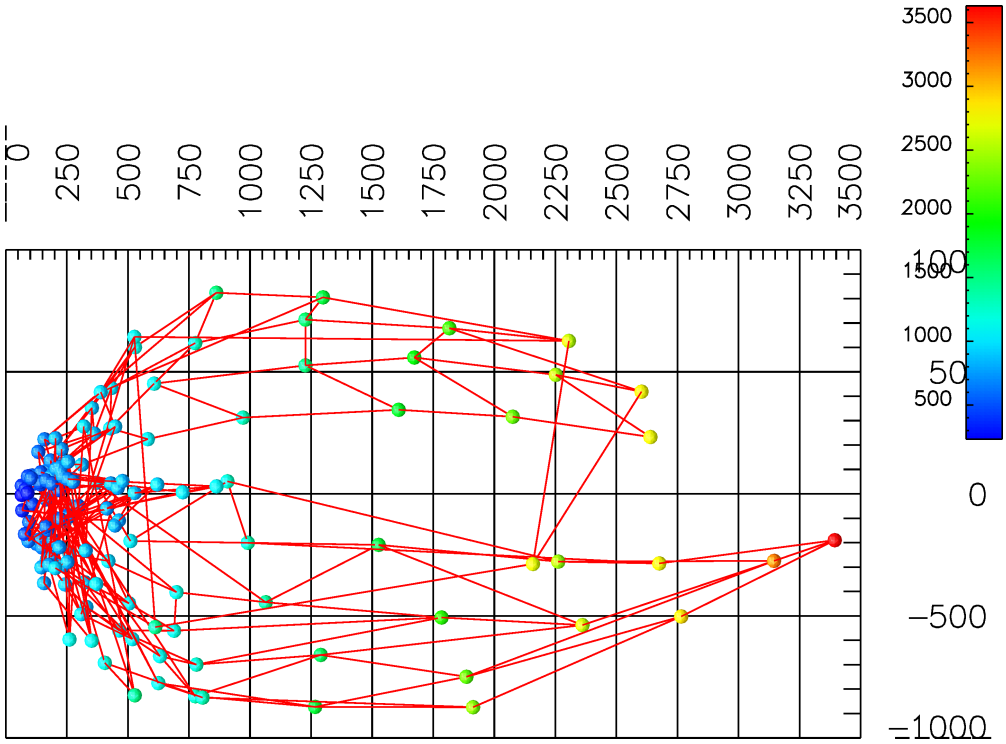


(b) First principal component

(c) Second principal component

(d) Third principal component

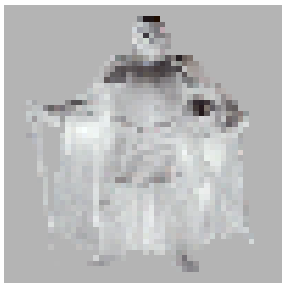
Figure D.18: The manifold KNIGHT_STANDING, 'mtfast'



(a) PCA Plot

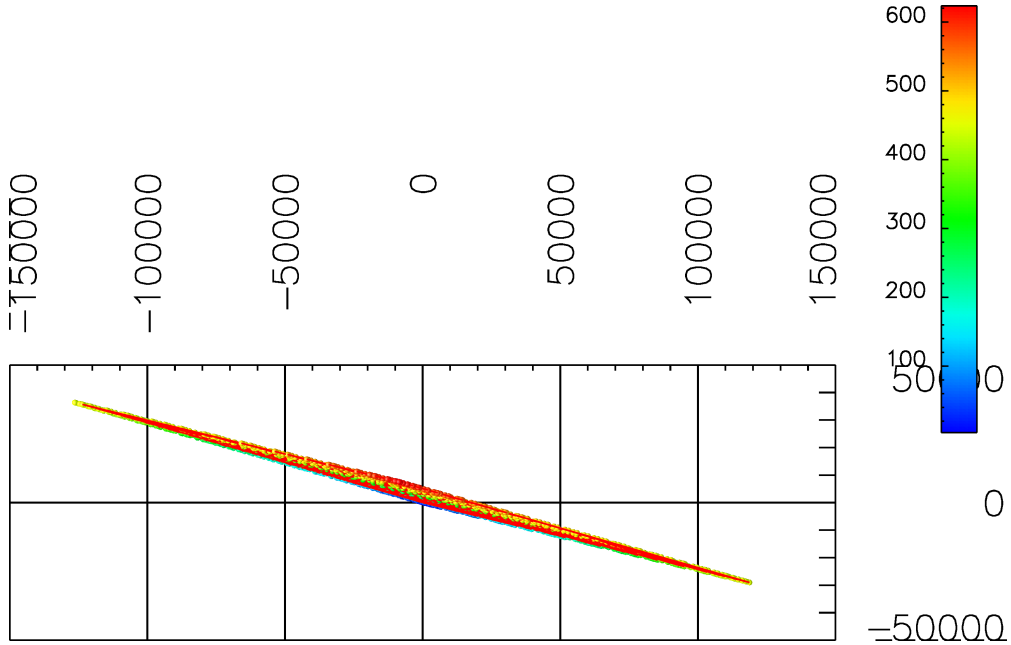
(b) First principal component

(c) Second principal component

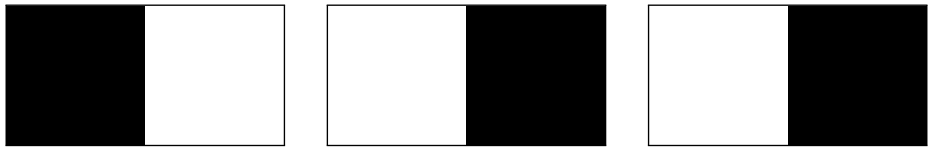


(d) Third principal component

Figure D.19: The manifold KNIGHT_STANDING, 'impca'



(a) PCA Plot

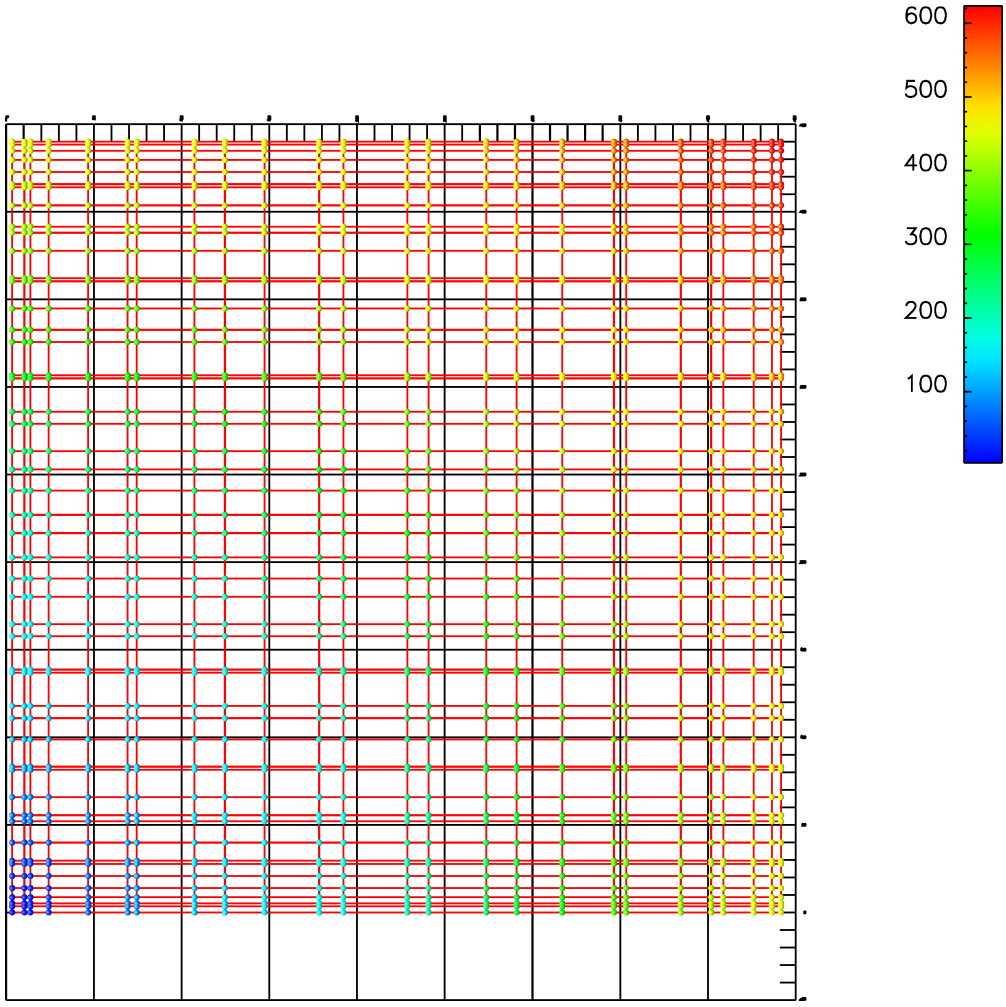


(b) First principal component

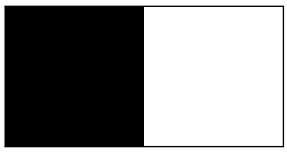
(c) Second principal component

(d) Third principal component

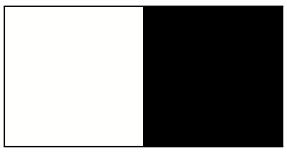
Figure D.20: The manifold Sinecos, 'mtfast'



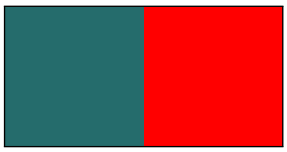
(a) PCA Plot



(b) First principal component

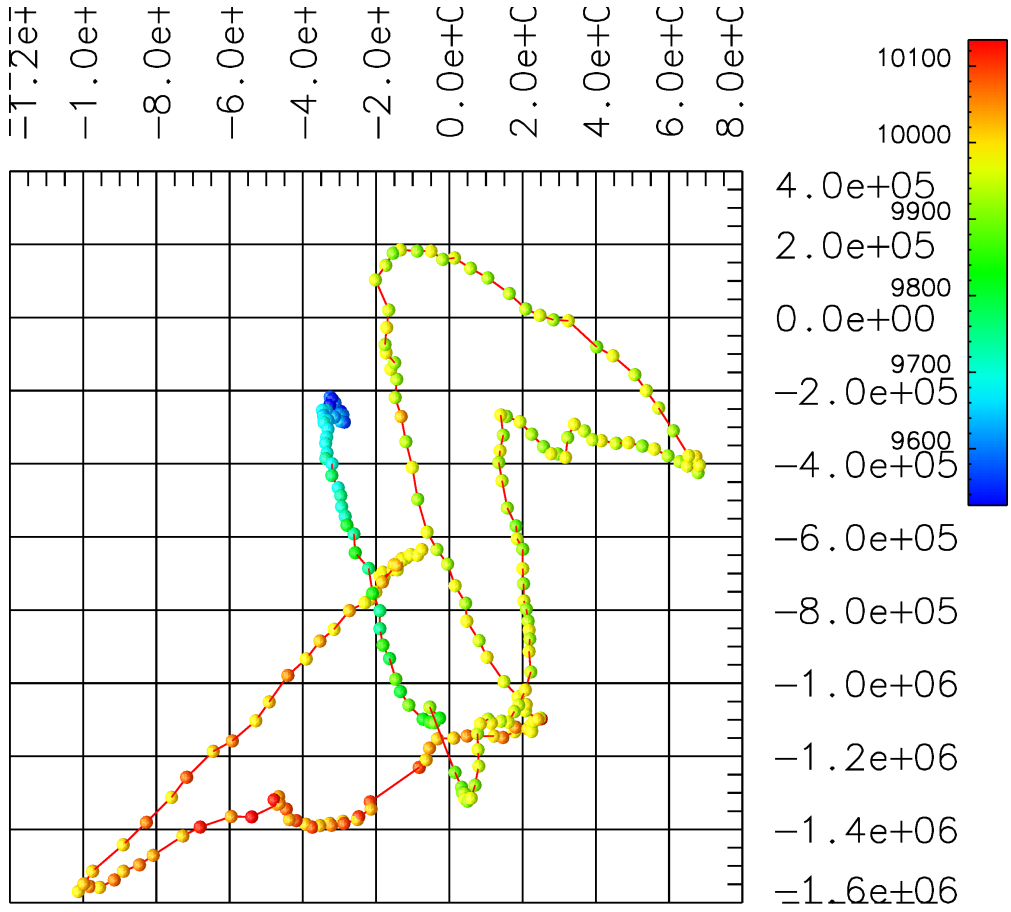


(c) Second principal component

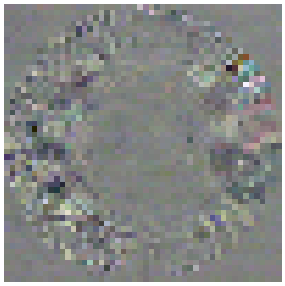


(d) Third principal component

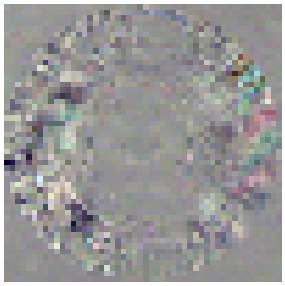
Figure D.21: The manifold Sinecos, 'impca'



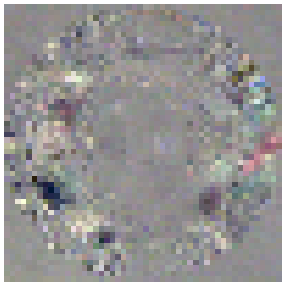
(a) PCA Plot



(b) First principal component

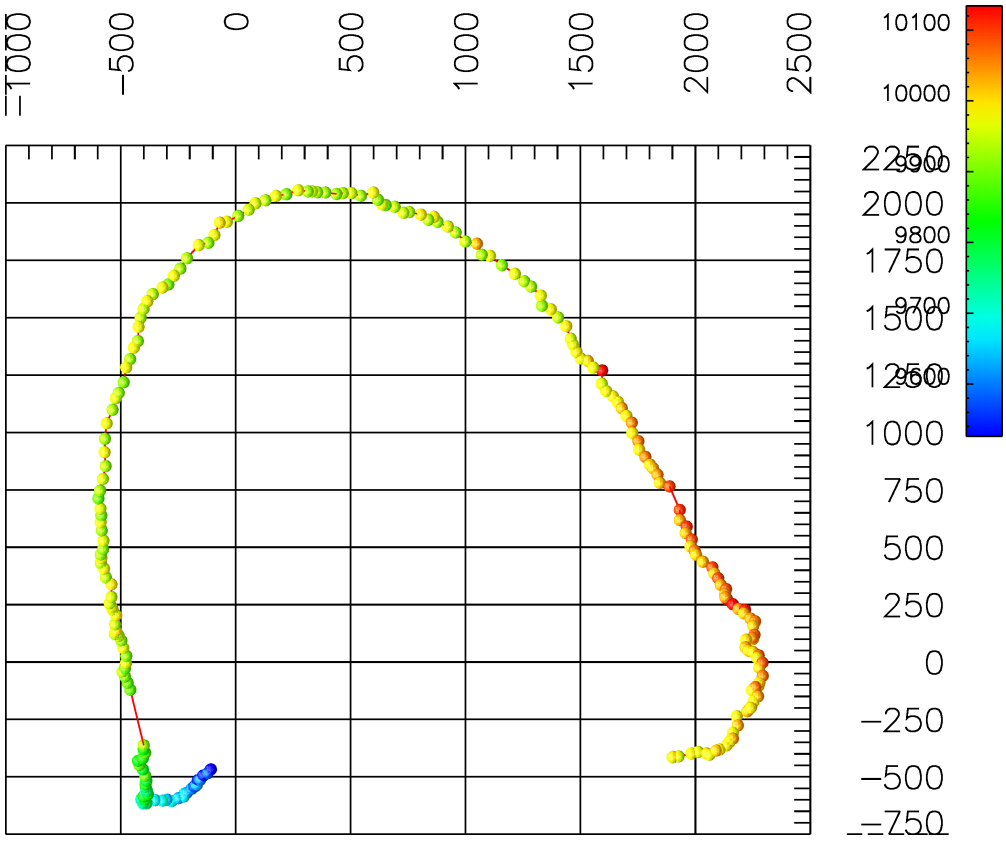


(c) Second principal component



(d) Third principal component

Figure D.22: The manifold STRAIGHT_1, 'mtfast'



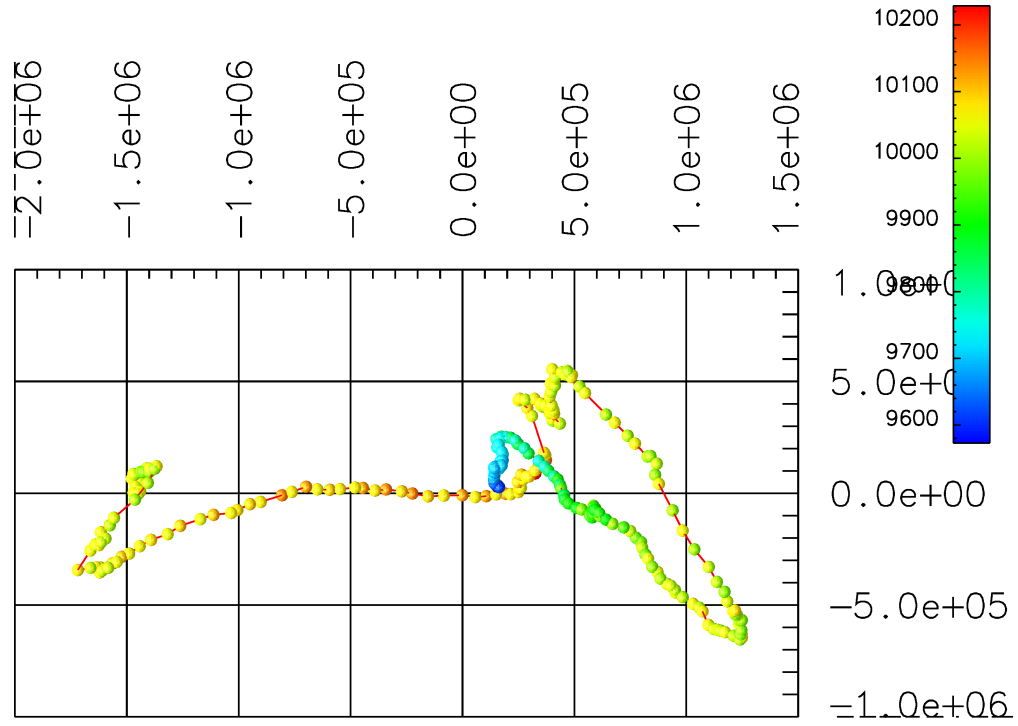
(a) PCA Plot

(b) First principal component

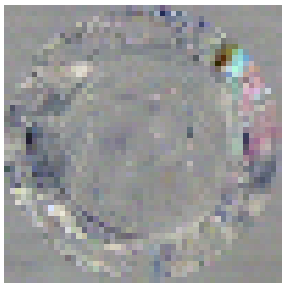
(c) Second principal component

(d) Third principal component

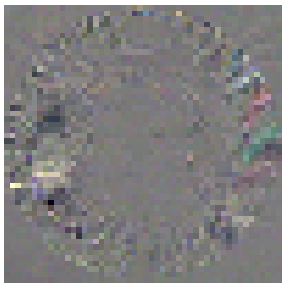
Figure D.23: The manifold STRAIGHT_1, 'impca'



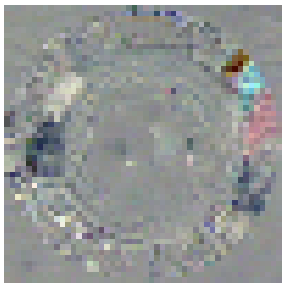
(a) PCA Plot



(b) First principal component

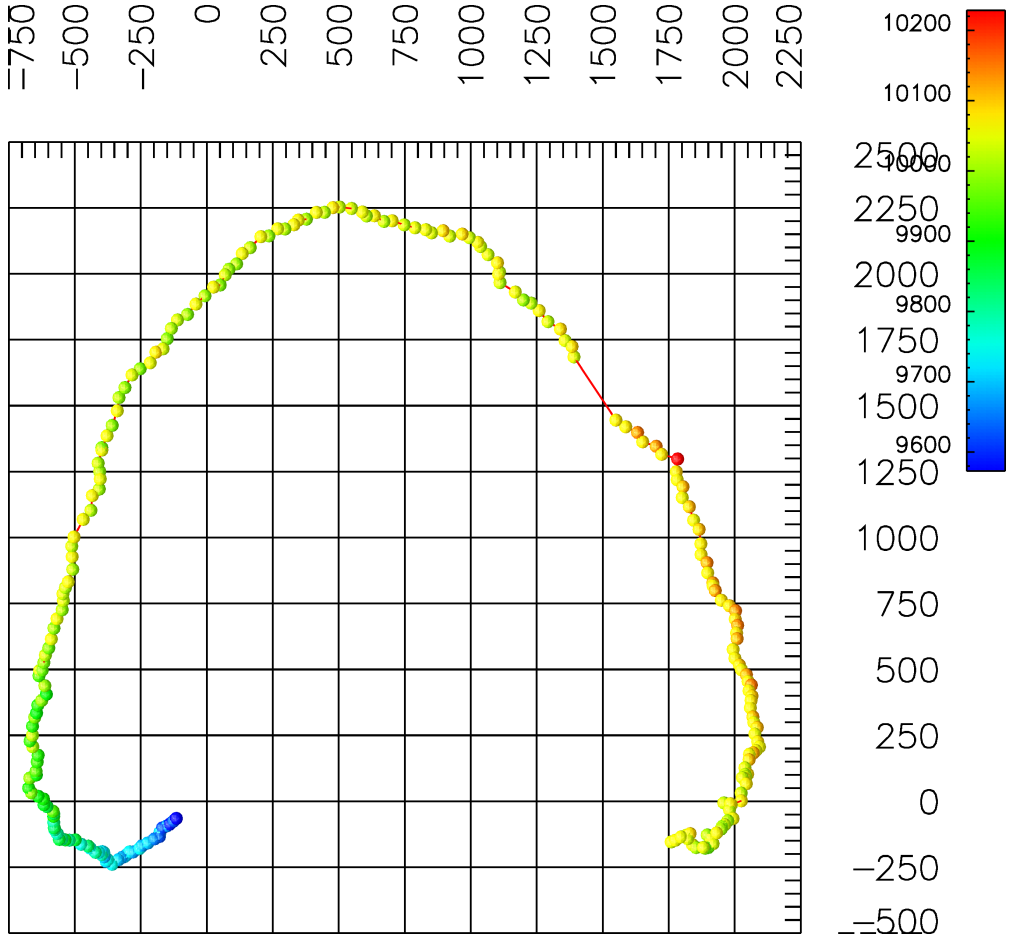


(c) Second principal component



(d) Third principal component

Figure D.24: The manifold STRAIGHT_2, 'mtfast'



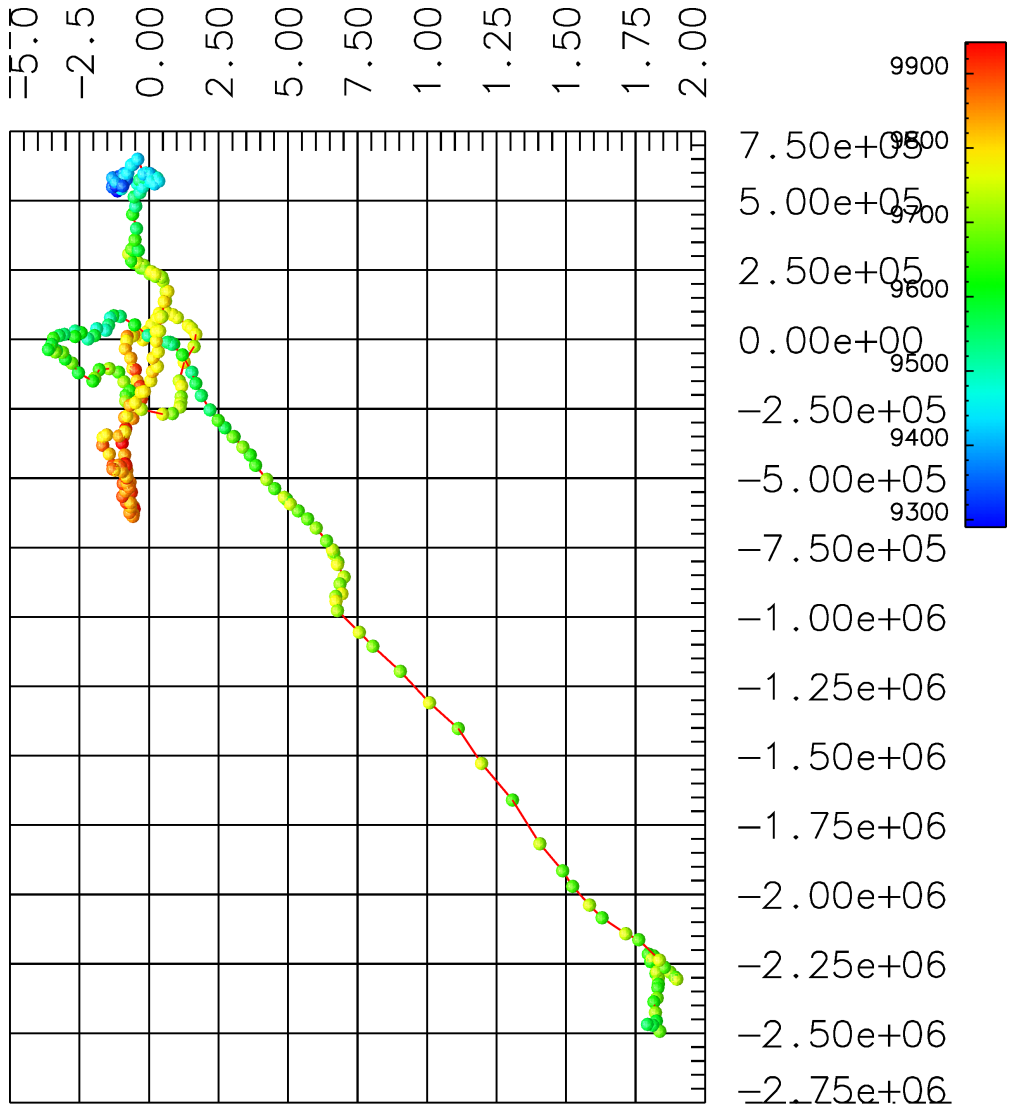
(a) PCA Plot

(b) First principal component

(c) Second principal component

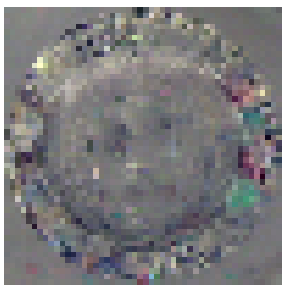
(d) Third principal component

Figure D.25: The manifold STRAIGHT_2, 'impca'

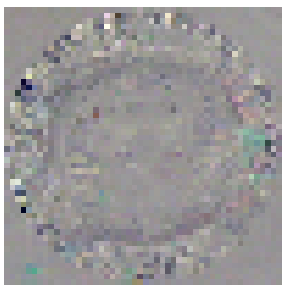


(a) PCA Plot

(b) First principal component

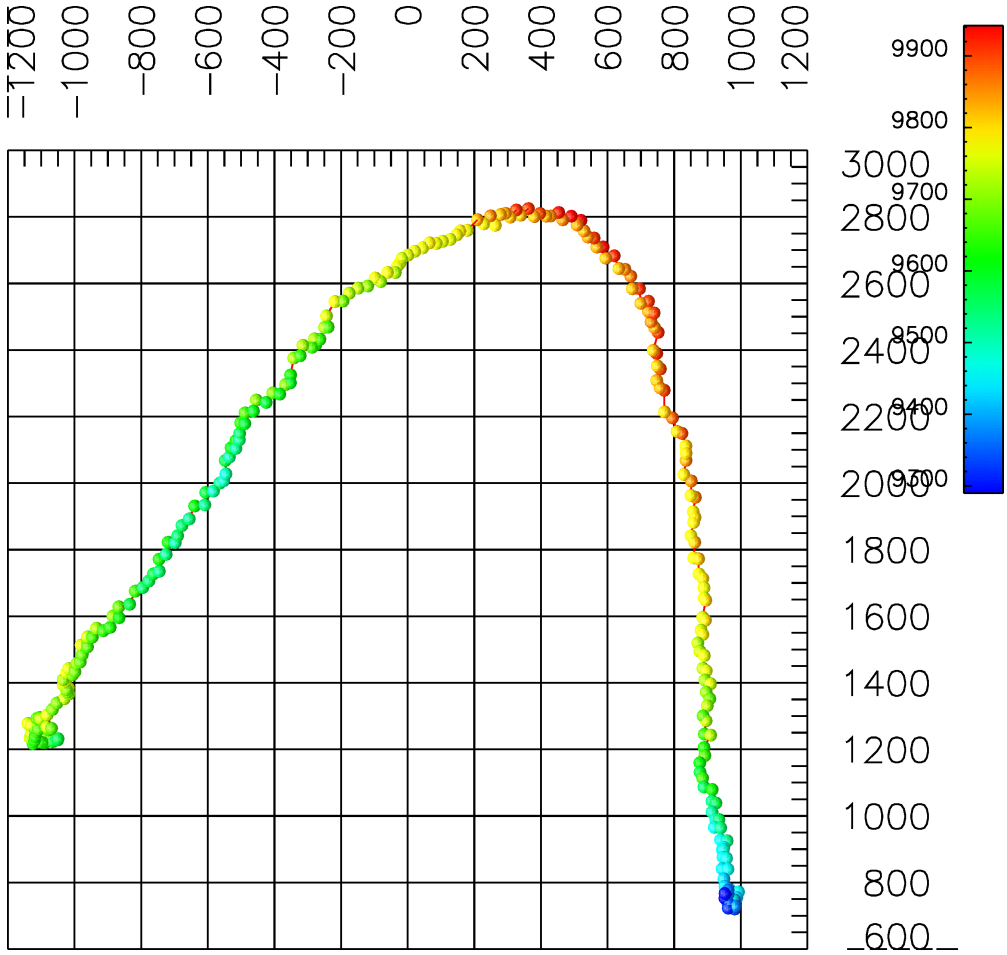


(c) Second principal component



(d) Third principal component

Figure D.26: The manifold STRAIGHT_3, 'mtfast'



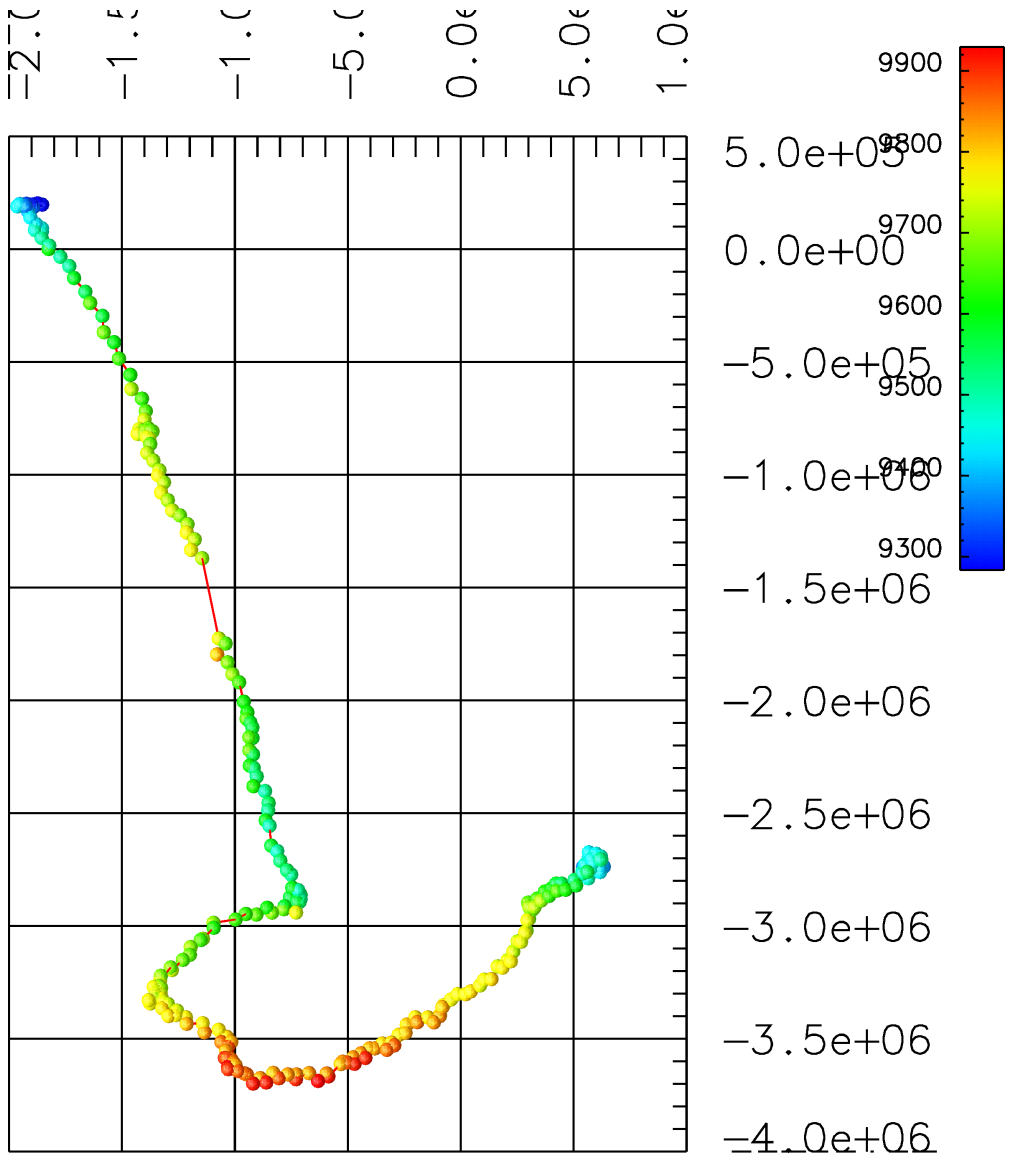
(a) PCA Plot

(b) First principal component

(c) Second principal component

(d) Third principal component

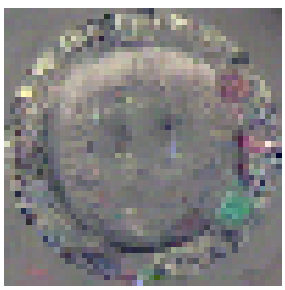
Figure D.27: The manifold STRAIGHT_3, 'impca'



(a) PCA Plot

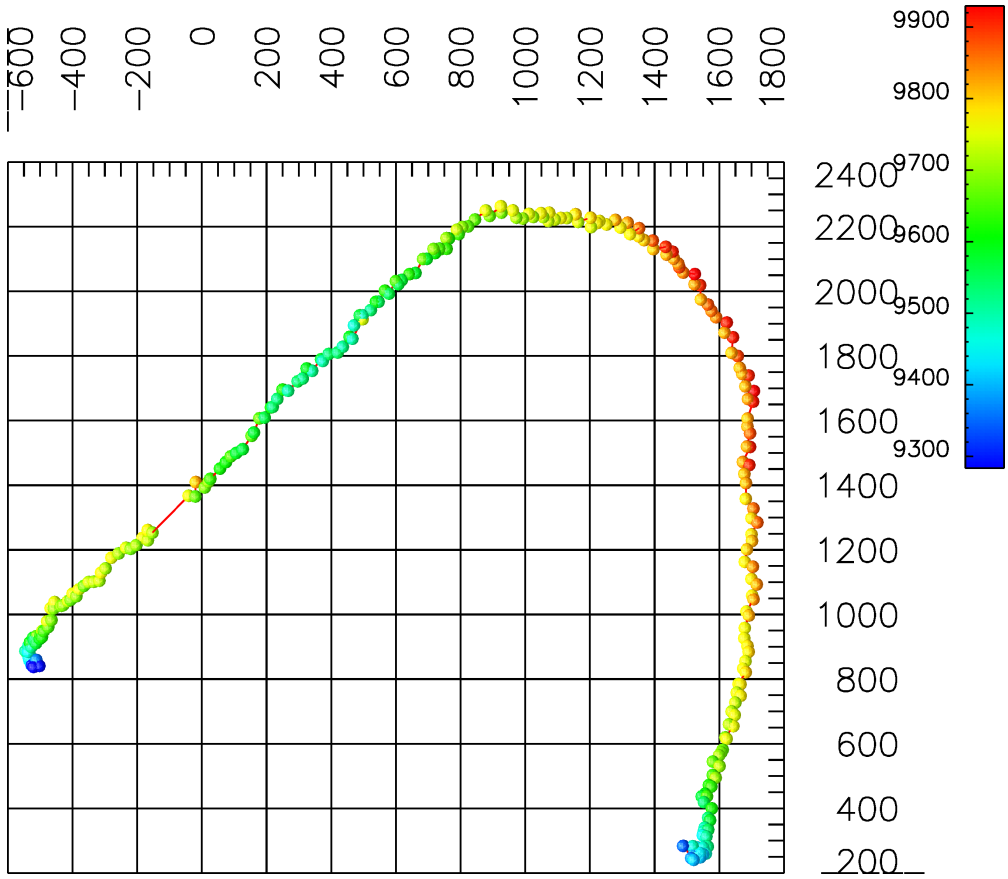
(b) First principal component

(c) Second principal component



(d) Third principal component

Figure D.28: The manifold STRAIGHT_4, 'mtfast'



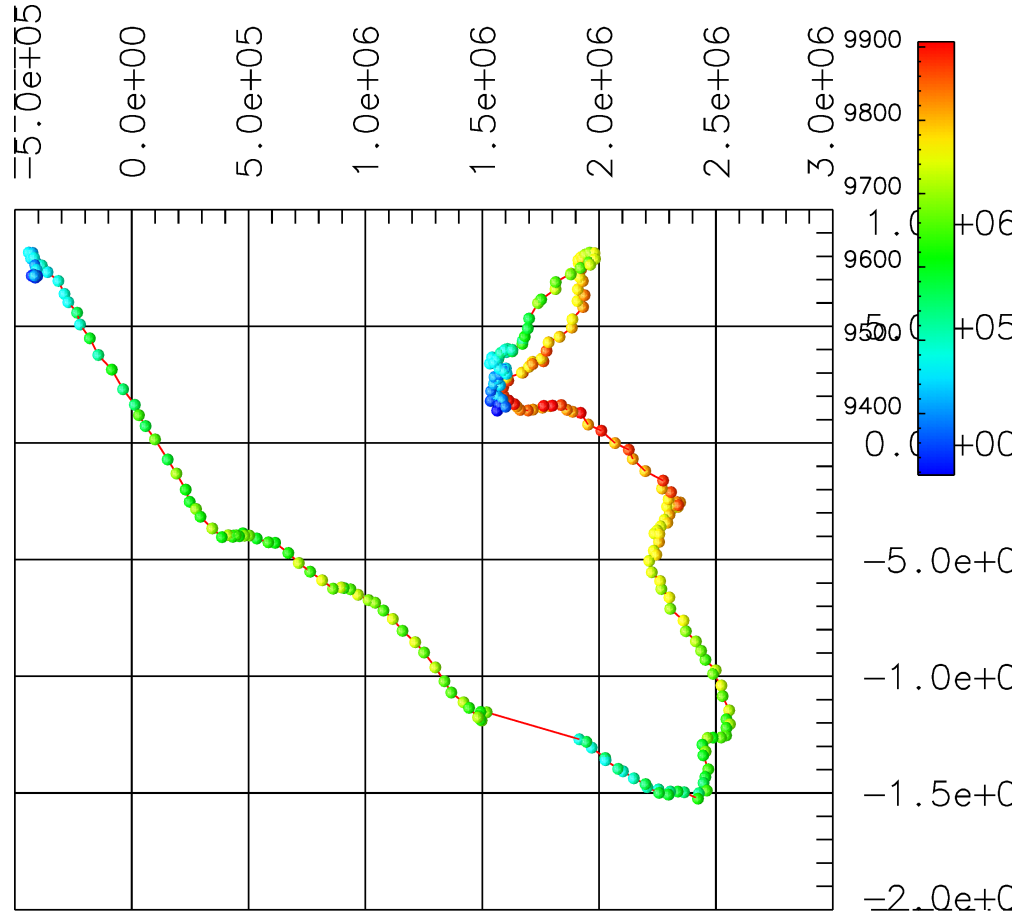
(a) PCA Plot

(b) First principal component

(c) Second principal component

(d) Third principal component

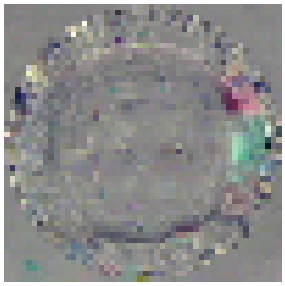
Figure D.29: The manifold Straight_4, 'impca'



(a) PCA Plot

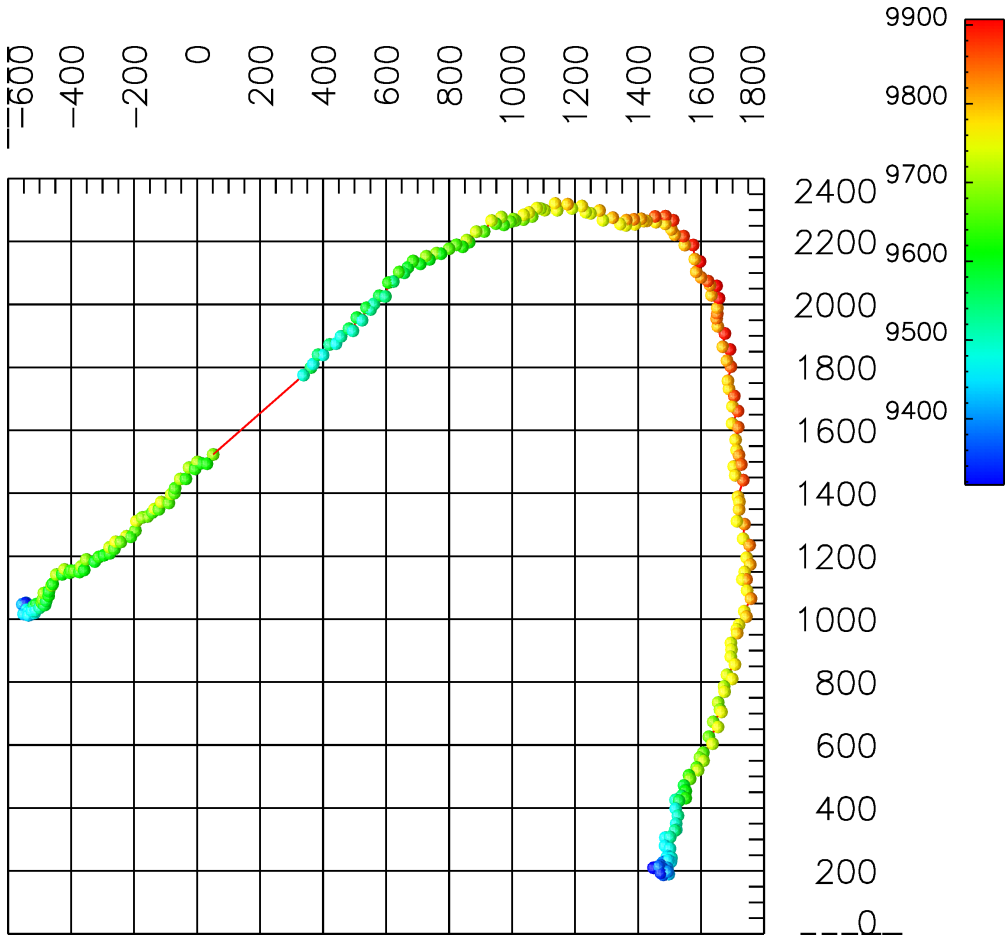
(b) First principal component

(c) Second principal component



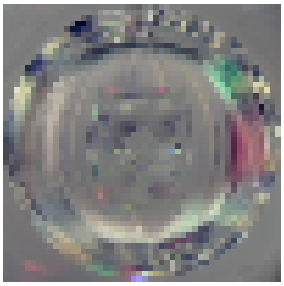
(d) Third principal component

Figure D.30: The manifold Straight_5, 'mtfast'



(a) PCA Plot

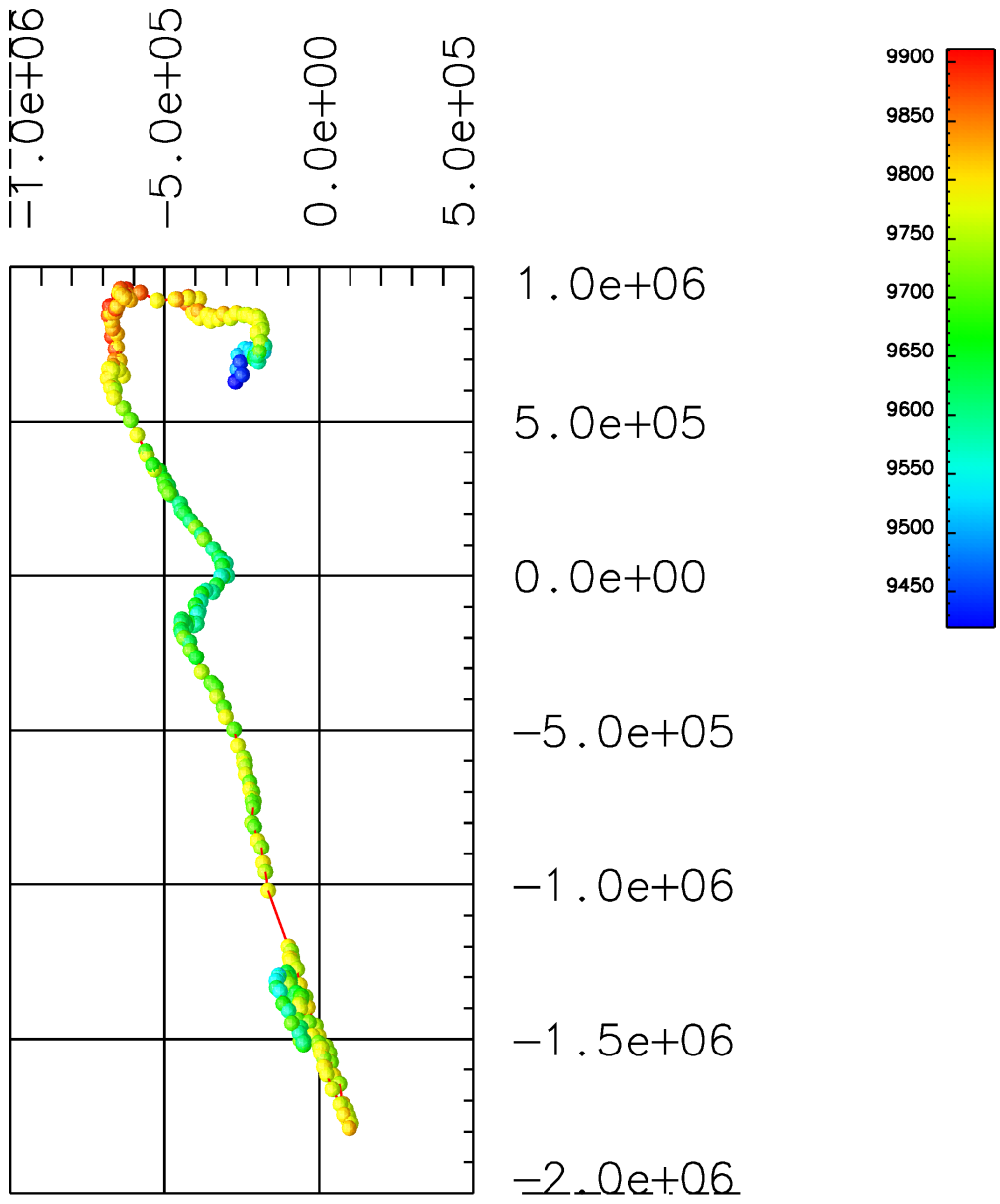
(b) First principal component



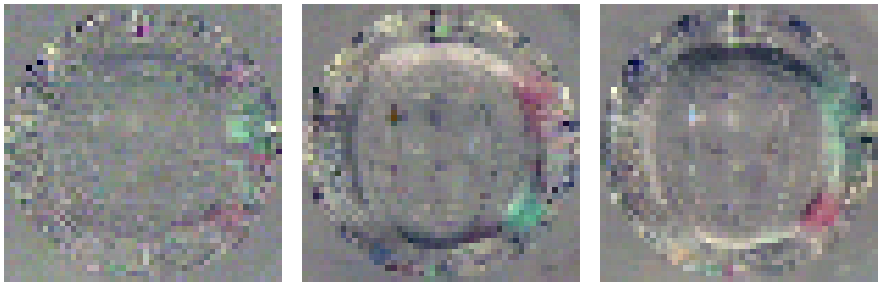
(c) Second principal component

(d) Third principal component

Figure D.31: The manifold Straight_5, 'impca'

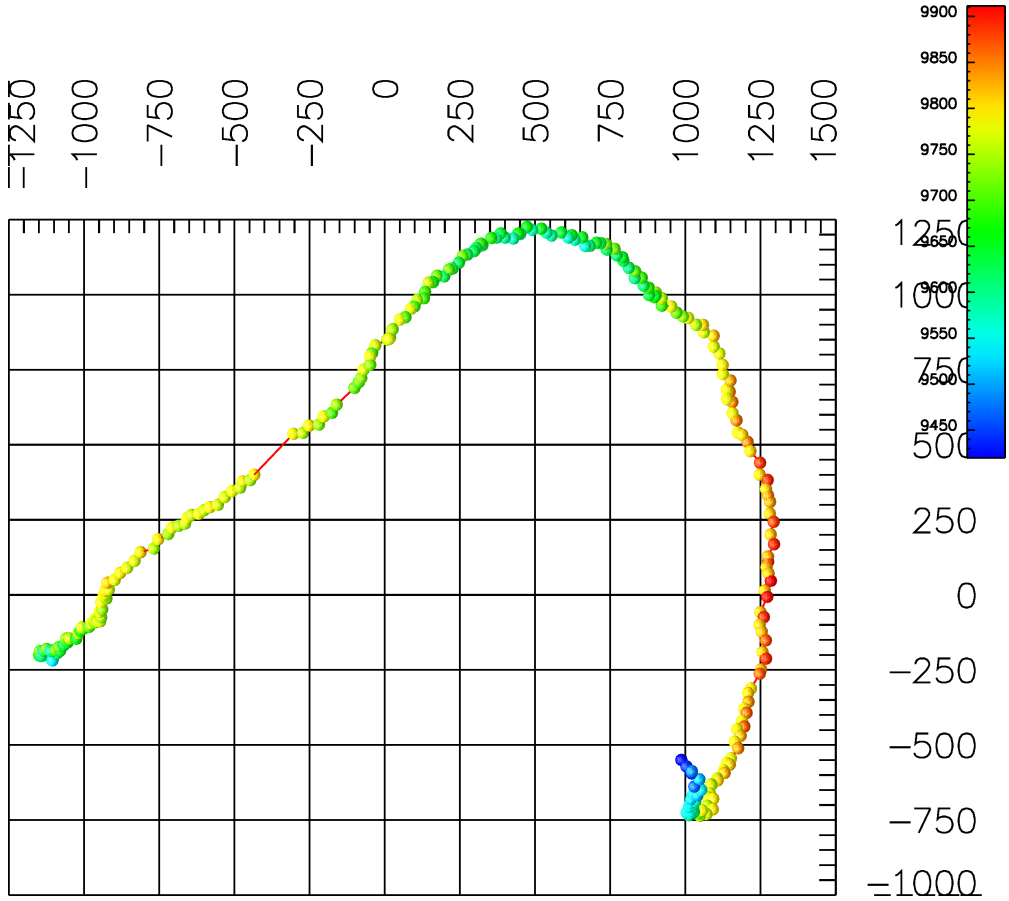


(a) PCA Plot



(b) First principal component (c) Second principal component (d) Third principal component

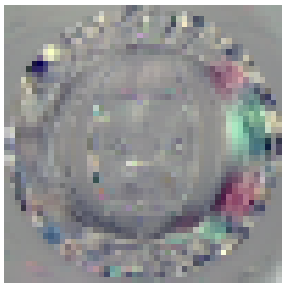
Figure D.32: The manifold STRAIGHT_6, 'mtfast'



(a) PCA Plot

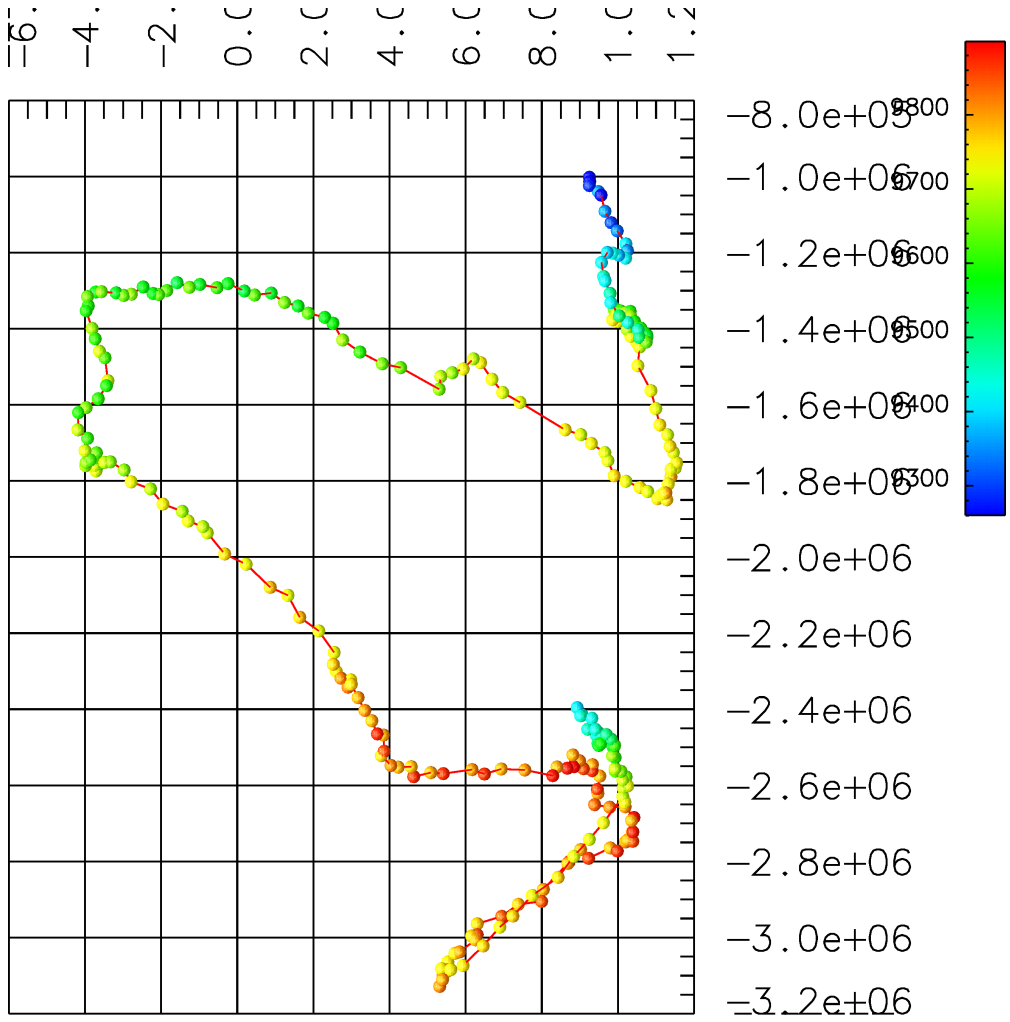
(b) First principal component

(c) Second principal component

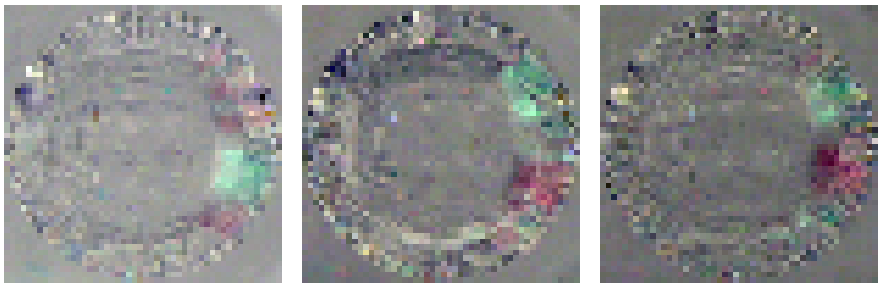


(d) Third principal component

Figure D.33: The manifold STRAIGHT_6, 'impca'



(a) PCA Plot

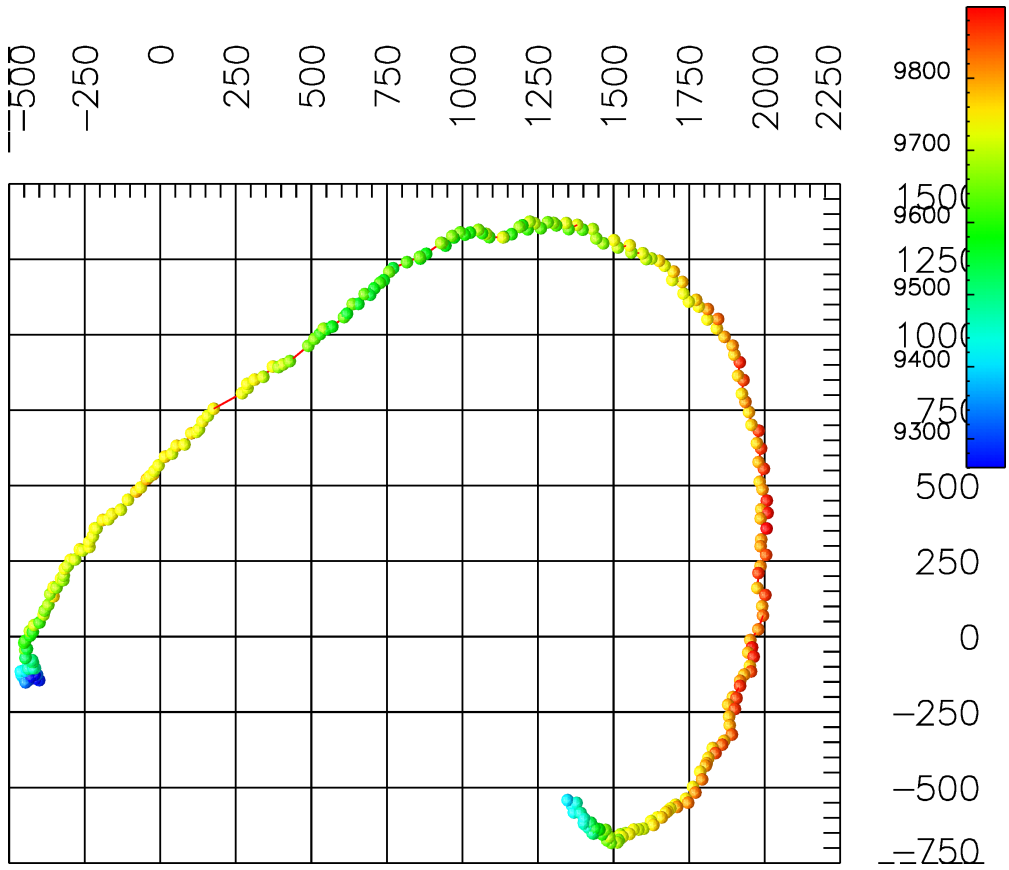


(b) First principal component

(c) Second principal component

(d) Third principal component

Figure D.34: The manifold STRAIGHT_7, 'mtfast'



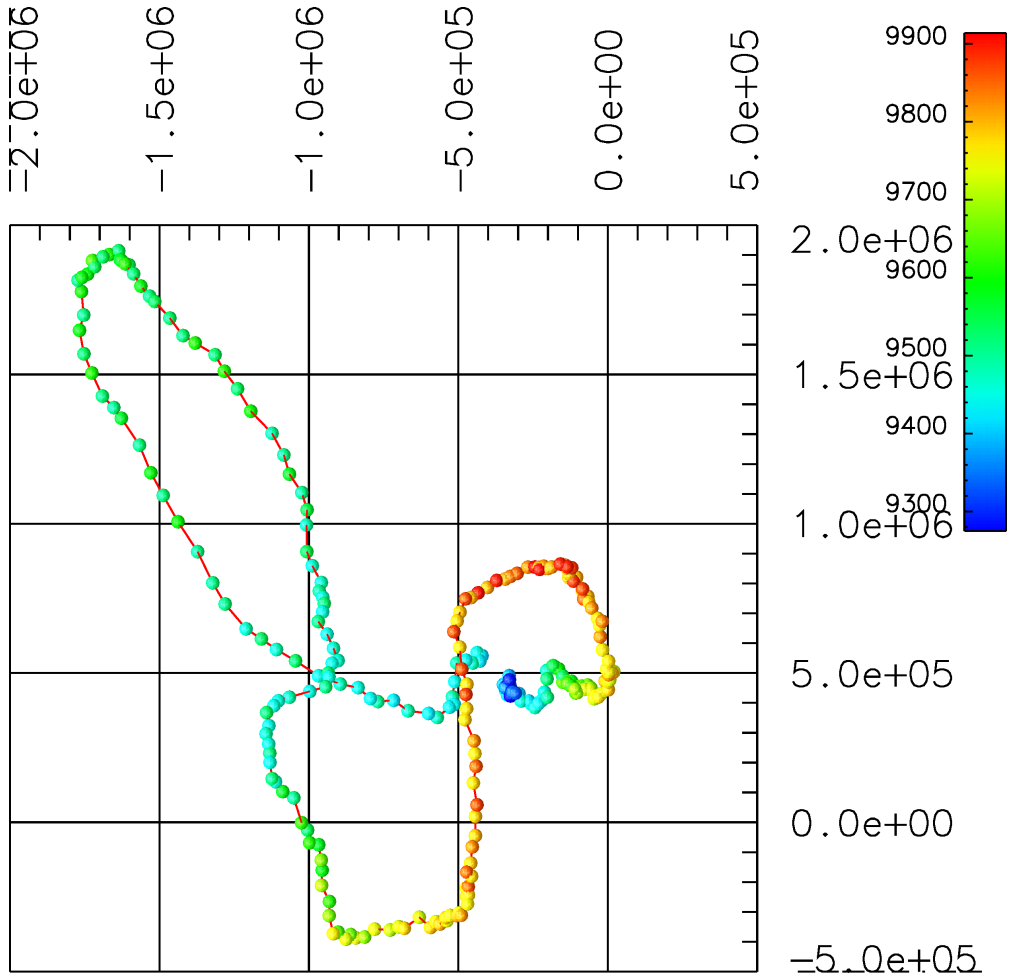
(a) PCA Plot

(b) First principal component

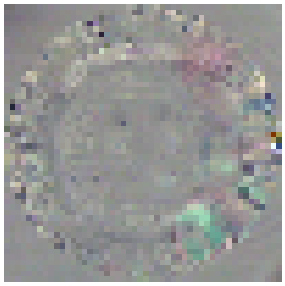
(c) Second principal component

(d) Third principal component

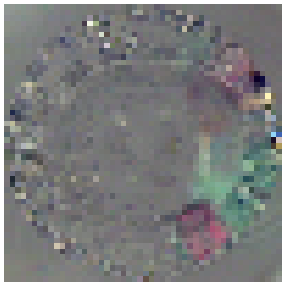
Figure D.35: The manifold STRAIGHT_7, 'impca'



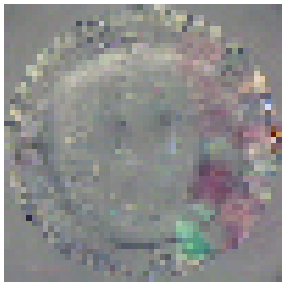
(a) PCA Plot



(b) First principal component

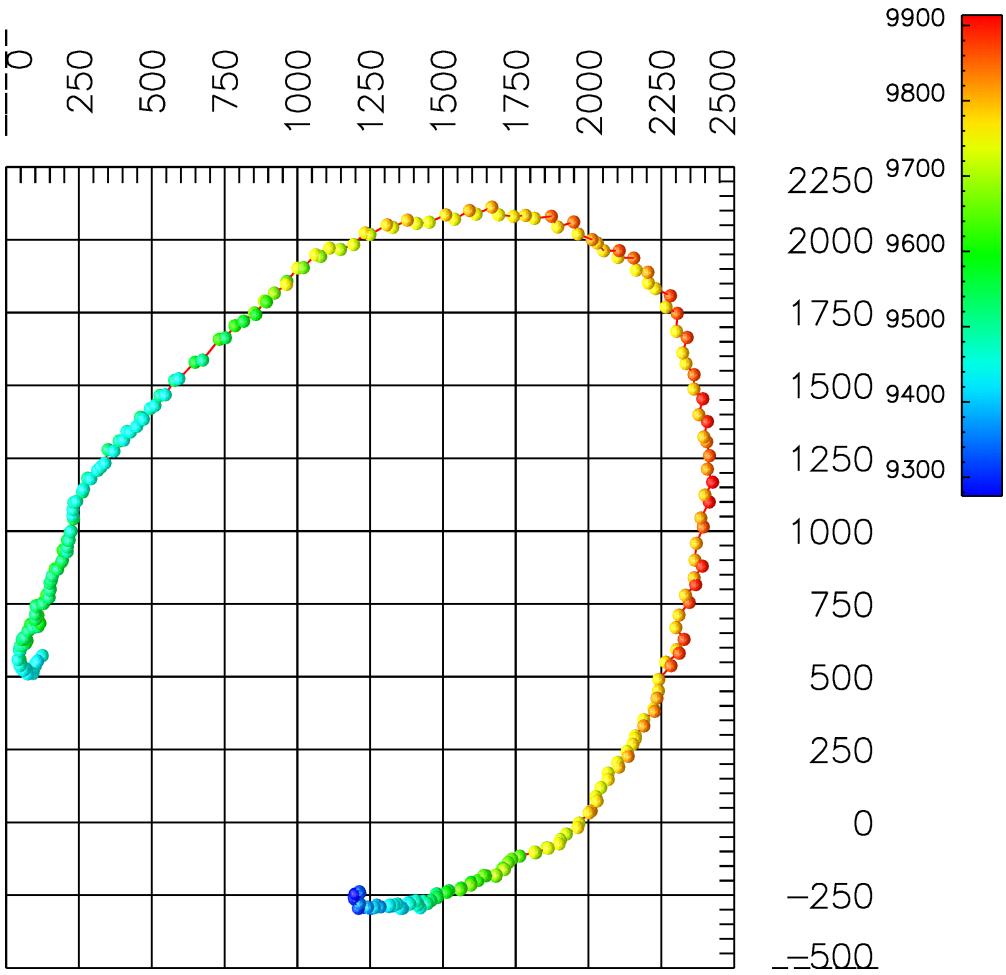


(c) Second principal component



(d) Third principal component

Figure D.36: The manifold STRAIGHT_8, 'mtfast'



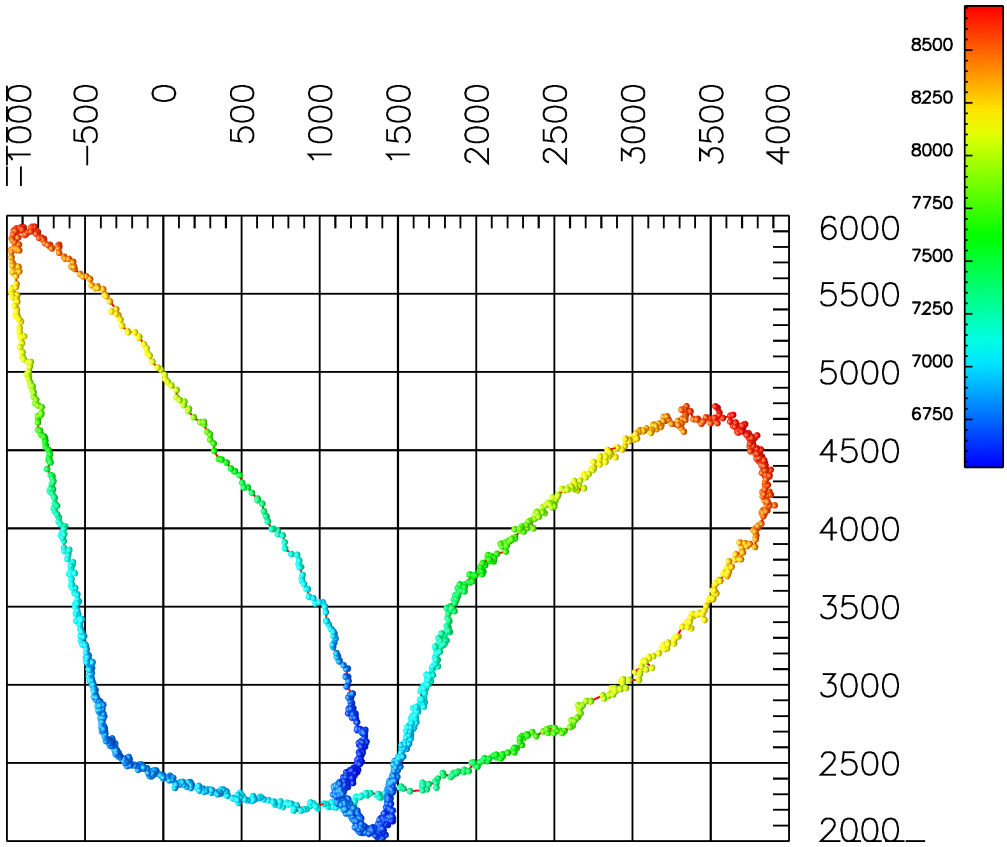
(a) PCA Plot

(b) First principal component

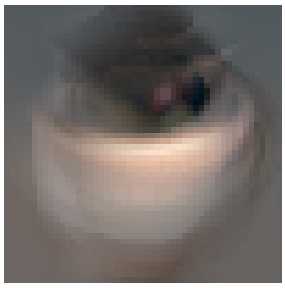
(c) Second principal component

(d) Third principal component

Figure D.37: The manifold Straight_8



(a) PCA Plot



(b) First principal component

(c) Second principal component

(d) Third principal component

Figure D.38: The manifold WOODBOX, 'impca'

Appendix E

Linear Model

In this appendix we present our results from the linear model. In these tables, which are provided for each of the metrics we used to evaluate the results (Chapter 5), we state the average measured error between the ground truth and the output of the model, as well as the standard deviation of this error. Results are shown for all of the values of g (gap) we considered. Only one configuration was used for the linear model.

E.1 Summary tables

Table E.1: Summary of the results from the Linear model (Metric=taxi)

Linear results (taxi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DTEAPOT	$\mu = 18384.8$, $\sigma = 11910.4$	$\mu = 34783.6$, $\sigma = 11351.7$	$\mu = 48977.6$, $\sigma = 13664$	$\mu = 66661.3$, $\sigma = 18485.1$	$\mu = 74373.7$, $\sigma = 21046.1$	$\mu = 97163.7$, $\sigma = 27752.7$	DNF
2DWOODBOX	$\mu = 41640.4$, $\sigma = 34029.2$	$\mu = 69816.9$, $\sigma = 42939.9$	$\mu = 91003.9$, $\sigma = 54883$	$\mu = 110078$, $\sigma = 64240.2$	$\mu = 109165$, $\sigma = 61950.9$	$\mu = 97657.4$, $\sigma = 33476.9$	DNF
BMNOISE	$\mu = 84086.9$, $\sigma = 88649.9$	$\mu = 139717$, $\sigma = 80687.4$	$\mu = 152795$, $\sigma = 72034.4$	DNF	DNF	DNF	DNF
BRUSH	$\mu = 6410.65$, $\sigma = 6992.02$	$\mu = 13595.8$, $\sigma = 8331.45$	$\mu = 19087.7$, $\sigma = 8678.65$	$\mu = 28574.5$, $\sigma = 10464.7$	$\mu = 31616.7$, $\sigma = 11786.8$	$\mu = 51514.7$, $\sigma = 18454$	$\mu = 79301.7$, $\sigma = 32431.8$
CHESSBOARD	$\mu = 178171$, $\sigma = 122335$	$\mu = 502276$, $\sigma = 170744$	$\mu = 834945$, $\sigma = 186673$	$\mu = 955681$, $\sigma = 341264$	$\mu = 951417$, $\sigma = 264233$	DNF	DNF
CIRCLE_3	$\mu = 20209.2$, $\sigma = 20252.3$	$\mu = 43433.2$, $\sigma = 22732.1$	$\mu = 62331.9$, $\sigma = 24670.3$	$\mu = 83496.4$, $\sigma = 27897.1$	$\mu = 90707.1$, $\sigma = 28718.9$	$\mu = 112520$, $\sigma = 29384.3$	$\mu = 128495$, $\sigma = 25878.9$
CIRCLE_4	$\mu = 20398.4$, $\sigma = 20470$	$\mu = 44203.7$, $\sigma = 23174.7$	$\mu = 63610$, $\sigma = 25201.6$	$\mu = 86181.4$, $\sigma = 28663.8$	$\mu = 93362.6$, $\sigma = 29666.9$	$\mu = 113643$, $\sigma = 29303.4$	$\mu = 131040$, $\sigma = 25790.2$
CIRCLE_5	$\mu = 21444.3$, $\sigma = 21516.9$	$\mu = 45957.8$, $\sigma = 23974$	$\mu = 64442.6$, $\sigma = 25128.5$	$\mu = 82997.9$, $\sigma = 26000.2$	$\mu = 88729.2$, $\sigma = 25830.2$	$\mu = 106283$, $\sigma = 25116.1$	$\mu = 118048$, $\sigma = 21494.9$
EXPT03	$\mu = 43380$, $\sigma = 34308.5$	$\mu = 64855.5$, $\sigma = 32955$	DNF	DNF	DNF	DNF	DNF
FACES	$\mu = 169092$, $\sigma = 121660$	$\mu = 241424$, $\sigma = 84076.1$	DNF	DNF	DNF	DNF	DNF
IDRIS_CIRCLE	$\mu = 10913.4$, $\sigma = 11080.2$	$\mu = 24250.9$, $\sigma = 13146.2$	$\mu = 36475.3$, $\sigma = 15168.6$	$\mu = 52074.6$, $\sigma = 18635.8$	$\mu = 58207.9$, $\sigma = 19987.9$	$\mu = 79288.3$, $\sigma = 24761.4$	$\mu = 103762$, $\sigma = 29716.3$
IDRIS_FIG8	$\mu = 10600.6$, $\sigma = 10894.7$	$\mu = 24245.2$, $\sigma = 14095.1$	$\mu = 36137.4$, $\sigma = 17176$	$\mu = 51067.7$, $\sigma = 22094.3$	$\mu = 56710.5$, $\sigma = 24070$	$\mu = 76330$, $\sigma = 30318$	$\mu = 100910$, $\sigma = 37494.7$
IDRIS_STRAIGHT	$\mu = 7523.94$, $\sigma = 7480.81$	$\mu = 14091.6$, $\sigma = 8049.89$	$\mu = 17821.5$, $\sigma = 8460.25$	$\mu = 22673.3$, $\sigma = 9928.79$	$\mu = 24608.6$, $\sigma = 11132.8$	$\mu = 32199.3$, $\sigma = 13929.6$	$\mu = 40256.5$, $\sigma = 16042.2$
KNIGHT_FIGHTING	$\mu = 9764.61$, $\sigma = 7673.71$	$\mu = 13531.7$, $\sigma = 5937.29$	DNF	DNF	DNF	DNF	DNF
KNIGHT_KNEELING	$\mu = 13610.3$, $\sigma = 10314.5$	$\mu = 18463.3$, $\sigma = 7038$	DNF	DNF	DNF	DNF	DNF

Continued on next page...

Table E.1 – Continued

Linear results (taxi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
KNIGHT_STANDING	$\mu = 17395.7$, $\sigma = 15286.6$	$\mu = 21372.3$, $\sigma = 10961.8$	DNF	DNF	DNF	DNF	DNF
SINECOS	$\mu = 27.3061$, $\sigma = 92.9795$	$\mu = 105.326$, $\sigma = 151.03$	$\mu = 247.831$, $\sigma = 146.593$	$\mu = 369.004$, $\sigma = 203.062$	$\mu = 387.24$, $\sigma = 200.76$	DNF	DNF
STRAIGHT_1	$\mu = 10960.9$, $\sigma = 11036.6$	$\mu = 22163.4$, $\sigma = 11839$	$\mu = 31621.4$, $\sigma = 12040$	$\mu = 44827.3$, $\sigma = 15085.4$	$\mu = 50566.1$, $\sigma = 16743.1$	$\mu = 72661.7$, $\sigma = 23417.4$	$\mu = 97716.8$, $\sigma = 30546.4$
STRAIGHT_2	$\mu = 10730.4$, $\sigma = 10792.2$	$\mu = 21248.3$, $\sigma = 10875$	$\mu = 30017.6$, $\sigma = 11299.4$	$\mu = 40948.7$, $\sigma = 12695.3$	$\mu = 45926.1$, $\sigma = 14178.1$	$\mu = 63379.5$, $\sigma = 18521.5$	$\mu = 84392.6$, $\sigma = 24574.1$
STRAIGHT_3	$\mu = 10076.2$, $\sigma = 10156.8$	$\mu = 19014.2$, $\sigma = 9678.97$	$\mu = 25469.7$, $\sigma = 9167.38$	$\mu = 33975.9$, $\sigma = 10088.7$	$\mu = 37498.7$, $\sigma = 10640.5$	$\mu = 48859.2$, $\sigma = 13055.5$	$\mu = 67477.3$, $\sigma = 18567.2$
STRAIGHT_4	$\mu = 10123.3$, $\sigma = 10221.7$	$\mu = 19234.9$, $\sigma = 9863.09$	$\mu = 25727.7$, $\sigma = 9353.28$	$\mu = 34402.3$, $\sigma = 10167.1$	$\mu = 37513.8$, $\sigma = 10688.3$	$\mu = 50394.5$, $\sigma = 13623.7$	$\mu = 69387.8$, $\sigma = 19875.6$
STRAIGHT_5	$\mu = 10210.3$, $\sigma = 10344.7$	$\mu = 19539.8$, $\sigma = 10250.4$	$\mu = 25938.8$, $\sigma = 9380.08$	$\mu = 34742.5$, $\sigma = 10558.9$	$\mu = 39266.4$, $\sigma = 11896.1$	$\mu = 51688.5$, $\sigma = 14123.4$	$\mu = 72015.5$, $\sigma = 20836.1$
STRAIGHT_6	$\mu = 10332.9$, $\sigma = 10422.4$	$\mu = 19699.5$, $\sigma = 10053.8$	$\mu = 26492.5$, $\sigma = 9494.48$	$\mu = 35688$, $\sigma = 10518$	$\mu = 39450.1$, $\sigma = 11127.8$	$\mu = 52451.9$, $\sigma = 14210.3$	$\mu = 71967.3$, $\sigma = 19922.5$
STRAIGHT_7	$\mu = 10284.1$, $\sigma = 10360.1$	$\mu = 19562.6$, $\sigma = 9940.71$	$\mu = 26379.1$, $\sigma = 9504.61$	$\mu = 35653.4$, $\sigma = 10575.2$	$\mu = 38951.1$, $\sigma = 11196.7$	$\mu = 52633.5$, $\sigma = 14572.2$	$\mu = 73858.8$, $\sigma = 21499$
STRAIGHT_8	$\mu = 10285.6$, $\sigma = 10388$	$\mu = 19622.8$, $\sigma = 10087.6$	$\mu = 26551.3$, $\sigma = 9819.8$	$\mu = 36039.7$, $\sigma = 11374.7$	$\mu = 40219.8$, $\sigma = 12271.5$	$\mu = 54459.4$, $\sigma = 16306.6$	$\mu = 80190.2$, $\sigma = 25680.3$
WOODBOX	$\mu = 12754.4$, $\sigma = 13160$	$\mu = 26509.8$, $\sigma = 14148.8$	$\mu = 35066.4$, $\sigma = 13305.5$	$\mu = 42864.6$, $\sigma = 12941.2$	$\mu = 45572.9$, $\sigma = 13070.5$	$\mu = 56293.1$, $\sigma = 16021$	$\mu = 71813.9$, $\sigma = 22790.9$

Table E.2: Summary of the results from the Linear model (Metric=pdiff)

Linear results (pdiff)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DTEAPOT	$\mu = 214.197$, $\sigma = 149.665$	$\mu = 328.903$, $\sigma = 104.413$	$\mu = 363.493$, $\sigma = 83.4608$	$\mu = 390.226$, $\sigma = 72.5602$	$\mu = 395.383$, $\sigma = 74.0811$	$\mu = 421.677$, $\sigma = 83.2806$	DNF
2DWOODBOX	$\mu = 633.72$, $\sigma = 430.302$	$\mu = 857.004$, $\sigma = 304.347$	$\mu = 945.567$, $\sigma = 265.403$	$\mu = 1081.88$, $\sigma = 246.668$	$\mu = 1099.45$, $\sigma = 275.714$	$\mu = 1341$, $\sigma = 147.19$	DNF
BMNOISE	$\mu = 1098.79$, $\sigma = 1158.24$	$\mu = 1825.56$, $\sigma = 1054.01$	$\mu = 1975.95$, $\sigma = 932.058$	DNF	DNF	DNF	DNF
BRUSH	$\mu = 51.6763$, $\sigma = 60.8966$	$\mu = 105.721$, $\sigma = 70.1272$	$\mu = 143.832$, $\sigma = 69.8973$	$\mu = 215.331$, $\sigma = 91.9048$	$\mu = 212.968$, $\sigma = 80.3218$	$\mu = 383.366$, $\sigma = 205.997$	$\mu = 483.535$, $\sigma = 169.603$
CHESSBOARD	$\mu = 232.903$, $\sigma = 159.925$	$\mu = 1018.71$, $\sigma = 302.14$	$\mu = 1500.06$, $\sigma = 395.999$	$\mu = 1249.26$, $\sigma = 446.097$	$\mu = 1517.04$, $\sigma = 435.572$	DNF	DNF
CIRCLE_3	$\mu = 651.401$, $\sigma = 657.507$	$\mu = 1115.61$, $\sigma = 567.565$	$\mu = 1326.47$, $\sigma = 454.717$	$\mu = 1501.73$, $\sigma = 366.605$	$\mu = 1551.75$, $\sigma = 341.696$	$\mu = 1664.21$, $\sigma = 269.958$	$\mu = 1746.9$, $\sigma = 208.451$
CIRCLE_4	$\mu = 659.544$, $\sigma = 665.007$	$\mu = 1131.82$, $\sigma = 574.502$	$\mu = 1348.02$, $\sigma = 460.069$	$\mu = 1529.94$, $\sigma = 369.559$	$\mu = 1577.11$, $\sigma = 343.625$	$\mu = 1682.95$, $\sigma = 269.096$	$\mu = 1773.21$, $\sigma = 207.957$
CIRCLE_5	$\mu = 675.002$, $\sigma = 679.974$	$\mu = 1148.89$, $\sigma = 582.719$	$\mu = 1356.05$, $\sigma = 462.051$	$\mu = 1517.61$, $\sigma = 364.388$	$\mu = 1559.2$, $\sigma = 336.933$	$\mu = 1658.84$, $\sigma = 261.735$	$\mu = 1732.08$, $\sigma = 202.326$
EXPT03	$\mu = 297.935$, $\sigma = 474.933$	$\mu = 389.98$, $\sigma = 528.023$	DNF	DNF	DNF	DNF	DNF
FACES	$\mu = 392.516$, $\sigma = 342.846$	$\mu = 588.44$, $\sigma = 266.084$	DNF	DNF	DNF	DNF	DNF
IDRIS_CIRCLE	$\mu = 150.169$, $\sigma = 151.964$	$\mu = 288.666$, $\sigma = 148.057$	$\mu = 375.518$, $\sigma = 132.139$	$\mu = 479.525$, $\sigma = 130.59$	$\mu = 520.712$, $\sigma = 134.219$	$\mu = 653.039$, $\sigma = 152.648$	$\mu = 805.543$, $\sigma = 176.406$
IDRIS_FIG8	$\mu = 137.174$, $\sigma = 138.699$	$\mu = 273.166$, $\sigma = 141.969$	$\mu = 362.037$, $\sigma = 134.963$	$\mu = 468.738$, $\sigma = 145.209$	$\mu = 506.802$, $\sigma = 153.059$	$\mu = 636.64$, $\sigma = 182.167$	$\mu = 788.632$, $\sigma = 214.889$
IDRIS_STRAIGHT	$\mu = 108.155$, $\sigma = 109.683$	$\mu = 208.65$, $\sigma = 110.605$	$\mu = 254.641$, $\sigma = 96.7542$	$\mu = 298.572$, $\sigma = 87.8889$	$\mu = 318.83$, $\sigma = 91.1785$	$\mu = 387.321$, $\sigma = 104.101$	$\mu = 458.564$, $\sigma = 129.794$
KNIGHT_FIGHTING	$\mu = 28.8889$, $\sigma = 25.0858$	$\mu = 50.6562$, $\sigma = 23.6083$	DNF	DNF	DNF	DNF	DNF
KNIGHT_KNEELING	$\mu = 47.5926$, $\sigma = 40.1376$	$\mu = 74.3646$, $\sigma = 31.7415$	DNF	DNF	DNF	DNF	DNF

Continued on next page...

Table E.2 – Continued

Linear results (pdiff)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
KNIGHT_STANDING	$\mu = 68.7778$, $\sigma = 56.9173$	$\mu = 113.125$, $\sigma = 45.5913$	DNF	DNF	DNF	DNF	DNF
SINECOS	$\mu = 0.0612245$, $\sigma = 0.243191$	$\mu = 0.245747$, $\sigma = 0.460239$	$\mu = 0.856633$, $\sigma = 0.604587$	$\mu = 0.999405$, $\sigma = 0.646686$	$\mu = 1.10947$, $\sigma = 0.63187$	DNF	DNF
STRAIGHT_1	$\mu = 624.82$, $\sigma = 630.266$	$\mu = 1043.91$, $\sigma = 531.893$	$\mu = 1211.45$, $\sigma = 418.354$	$\mu = 1347.65$, $\sigma = 329.629$	$\mu = 1388.64$, $\sigma = 307.38$	$\mu = 1520.19$, $\sigma = 256.081$	$\mu = 1610.98$, $\sigma = 226.129$
STRAIGHT_2	$\mu = 618.46$, $\sigma = 623.763$	$\mu = 1026.15$, $\sigma = 522.688$	$\mu = 1186.46$, $\sigma = 409.28$	$\mu = 1310.59$, $\sigma = 319.834$	$\mu = 1351.57$, $\sigma = 298.099$	$\mu = 1460.81$, $\sigma = 247.278$	$\mu = 1550.31$, $\sigma = 214.155$
STRAIGHT_3	$\mu = 610.579$, $\sigma = 616.127$	$\mu = 1015.06$, $\sigma = 517.81$	$\mu = 1161$, $\sigma = 401.454$	$\mu = 1278.71$, $\sigma = 313.734$	$\mu = 1317.83$, $\sigma = 291.696$	$\mu = 1406.96$, $\sigma = 230.761$	$\mu = 1511.91$, $\sigma = 200.949$
STRAIGHT_4	$\mu = 609.33$, $\sigma = 615.179$	$\mu = 1016.28$, $\sigma = 518.564$	$\mu = 1162.67$, $\sigma = 401.939$	$\mu = 1279.8$, $\sigma = 312.232$	$\mu = 1314.71$, $\sigma = 289.793$	$\mu = 1403.79$, $\sigma = 228.056$	$\mu = 1525.01$, $\sigma = 209.265$
STRAIGHT_5	$\mu = 613.72$, $\sigma = 619.354$	$\mu = 1020.85$, $\sigma = 521.382$	$\mu = 1167.08$, $\sigma = 403.875$	$\mu = 1286.54$, $\sigma = 315.354$	$\mu = 1329.92$, $\sigma = 296.014$	$\mu = 1420.44$, $\sigma = 235.283$	$\mu = 1544.63$, $\sigma = 214.444$
STRAIGHT_6	$\mu = 626.287$, $\sigma = 631.784$	$\mu = 1041.13$, $\sigma = 529.931$	$\mu = 1191.54$, $\sigma = 411.472$	$\mu = 1306.56$, $\sigma = 318.334$	$\mu = 1342.2$, $\sigma = 296.059$	$\mu = 1437.58$, $\sigma = 236.448$	$\mu = 1546.27$, $\sigma = 206.831$
STRAIGHT_7	$\mu = 626$, $\sigma = 631.331$	$\mu = 1041.69$, $\sigma = 531.657$	$\mu = 1190.8$, $\sigma = 411.599$	$\mu = 1314.95$, $\sigma = 321.856$	$\mu = 1355.93$, $\sigma = 301.048$	$\mu = 1457.31$, $\sigma = 244.038$	$\mu = 1547.24$, $\sigma = 215.447$
STRAIGHT_8	$\mu = 622.337$, $\sigma = 628.617$	$\mu = 1034.92$, $\sigma = 529.619$	$\mu = 1196.36$, $\sigma = 417.032$	$\mu = 1324.11$, $\sigma = 330.09$	$\mu = 1369.38$, $\sigma = 309.191$	$\mu = 1480$, $\sigma = 255.043$	$\mu = 1661.76$, $\sigma = 244.086$
WOODBOX	$\mu = 213.303$, $\sigma = 233.661$	$\mu = 413.322$, $\sigma = 250.379$	$\mu = 515.067$, $\sigma = 236.492$	$\mu = 582.729$, $\sigma = 223.318$	$\mu = 599.391$, $\sigma = 220.787$	$\mu = 646.035$, $\sigma = 216.625$	$\mu = 693.736$, $\sigma = 222.44$

Table E.3: Summary of the results from the Linear model (Metric=mi)

Linear results (mi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DWOODBOX	$\mu = 0.444636$, $\sigma = 0.252575$	$\mu = 0.591134$, $\sigma = 0.135355$	$\mu = 0.63432$, $\sigma = 0.0875132$	$\mu = 0.659189$, $\sigma = 0.0608471$	$\mu = 0.664493$, $\sigma = 0.0587706$	$\mu = 0.676635$, $\sigma = 0.0375959$	DNF
BMNOISE	$\mu = 0.299707$, $\sigma = 0.31592$	$\mu = 0.593804$, $\sigma = 0.343036$	$\mu = 0.65791$, $\sigma = 0.313793$	DNF	DNF	DNF	DNF
BRUSH	$\mu = 0.165183$, $\sigma = 0.164759$	$\mu = 0.276326$, $\sigma = 0.143376$	$\mu = 0.319764$, $\sigma = 0.113679$	$\mu = 0.355556$, $\sigma = 0.0914853$	$\mu = 0.363117$, $\sigma = 0.0842597$	$\mu = 0.396669$, $\sigma = 0.0735501$	$\mu = 0.415816$, $\sigma = 0.0660029$
CHESSBOARD	$\mu = 0.303706$, $\sigma = 0.188637$	$\mu = 0.71296$, $\sigma = 0.191329$	$\mu = 0.897861$, $\sigma = 0.173308$	$\mu = 0.89778$, $\sigma = 0.17326$	$\mu = 0.747077$, $\sigma = 0.129363$	DNF	DNF
CIRCLE_3	$\mu = 0.285798$, $\sigma = 0.27554$	$\mu = 0.478087$, $\sigma = 0.233057$	$\mu = 0.558964$, $\sigma = 0.183793$	$\mu = 0.613463$, $\sigma = 0.144167$	$\mu = 0.627023$, $\sigma = 0.133445$	$\mu = 0.66032$, $\sigma = 0.101954$	$\mu = 0.681855$, $\sigma = 0.0778441$
CIRCLE_4	$\mu = 0.285422$, $\sigma = 0.276689$	$\mu = 0.479032$, $\sigma = 0.234341$	$\mu = 0.560452$, $\sigma = 0.185757$	$\mu = 0.616465$, $\sigma = 0.143945$	$\mu = 0.629775$, $\sigma = 0.13422$	$\mu = 0.663035$, $\sigma = 0.102733$	$\mu = 0.682446$, $\sigma = 0.0776498$
CIRCLE_5	$\mu = 0.288294$, $\sigma = 0.280868$	$\mu = 0.484364$, $\sigma = 0.235348$	$\mu = 0.563233$, $\sigma = 0.187295$	$\mu = 0.616298$, $\sigma = 0.14452$	$\mu = 0.629911$, $\sigma = 0.133593$	$\mu = 0.661773$, $\sigma = 0.101904$	$\mu = 0.680585$, $\sigma = 0.0767816$
EXPT03	$\mu = 0.635105$, $\sigma = 0.281111$	$\mu = 0.743715$, $\sigma = 0.113082$	DNF	DNF	DNF	DNF	DNF
FACES	$\mu = 0.389117$, $\sigma = 0.246707$	$\mu = 0.507061$, $\sigma = 0.145317$	DNF	DNF	DNF	DNF	DNF
IDRIS_FIG8	$\mu = 0.21715$, $\sigma = 0.212965$	$\mu = 0.369653$, $\sigma = 0.18325$	$\mu = 0.436689$, $\sigma = 0.146053$	$\mu = 0.483995$, $\sigma = 0.112508$	$\mu = 0.495777$, $\sigma = 0.106042$	$\mu = 0.528531$, $\sigma = 0.0847421$	$\mu = 0.555659$, $\sigma = 0.0703119$
IDRIS_STRAIGHT	$\mu = 0.204806$, $\sigma = 0.196456$	$\mu = 0.334862$, $\sigma = 0.164907$	$\mu = 0.384684$, $\sigma = 0.133202$	$\mu = 0.420921$, $\sigma = 0.105465$	$\mu = 0.430142$, $\sigma = 0.0983317$	$\mu = 0.45897$, $\sigma = 0.0798707$	$\mu = 0.481348$, $\sigma = 0.0665042$
KNIGHT_FIGHTING	$\mu = 0.372034$, $\sigma = 0.250349$	$\mu = 0.537072$, $\sigma = 0.171381$	DNF	DNF	DNF	DNF	DNF
KNIGHT_KNEELING	$\mu = 0.498494$, $\sigma = 0.333298$	$\mu = 0.698817$, $\sigma = 0.217285$	DNF	DNF	DNF	DNF	DNF
KNIGHT_STANDING	$\mu = 0.541258$, $\sigma = 0.364163$	$\mu = 0.785575$, $\sigma = 0.241742$	DNF	DNF	DNF	DNF	DNF
SINECOS	DNF	$\mu = 0.00567108$, $\sigma = 0.0750927$	$\mu = 0.00832838$, $\sigma = 0.0908791$	$\mu = 0.00594884$, $\sigma = 0.076899$	$\mu = 0.00443787$, $\sigma = 0.0664694$	DNF	DNF

Continued on next page...

Table E.3 – Continued

Linear results (mi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
Straight_1	$\mu = 0.245663$, $\sigma = 0.241708$	$\mu = 0.411848$, $\sigma = 0.205316$	$\mu = 0.485097$, $\sigma = 0.164461$	$\mu = 0.539974$, $\sigma = 0.13153$	$\mu = 0.556495$, $\sigma = 0.122829$	$\mu = 0.601158$, $\sigma = 0.102231$	$\mu = 0.636043$, $\sigma = 0.0881994$
Straight_2	$\mu = 0.242485$, $\sigma = 0.240967$	$\mu = 0.408421$, $\sigma = 0.201299$	$\mu = 0.478868$, $\sigma = 0.162256$	$\mu = 0.530195$, $\sigma = 0.128544$	$\mu = 0.545436$, $\sigma = 0.119661$	$\mu = 0.586905$, $\sigma = 0.0984281$	$\mu = 0.619578$, $\sigma = 0.0836319$
Straight_3	$\mu = 0.245357$, $\sigma = 0.246341$	$\mu = 0.411991$, $\sigma = 0.205365$	$\mu = 0.481448$, $\sigma = 0.162843$	$\mu = 0.533219$, $\sigma = 0.128526$	$\mu = 0.548489$, $\sigma = 0.119489$	$\mu = 0.588934$, $\sigma = 0.0969848$	$\mu = 0.630529$, $\sigma = 0.087222$
Straight_4	$\mu = 0.24574$, $\sigma = 0.246749$	$\mu = 0.412992$, $\sigma = 0.205911$	$\mu = 0.482824$, $\sigma = 0.163413$	$\mu = 0.534915$, $\sigma = 0.129085$	$\mu = 0.548646$, $\sigma = 0.119666$	$\mu = 0.59176$, $\sigma = 0.0978471$	$\mu = 0.631467$, $\sigma = 0.087486$
Straight_5	$\mu = 0.246376$, $\sigma = 0.247419$	$\mu = 0.414711$, $\sigma = 0.206921$	$\mu = 0.48384$, $\sigma = 0.163746$	$\mu = 0.536247$, $\sigma = 0.129683$	$\mu = 0.553815$, $\sigma = 0.121751$	$\mu = 0.595062$, $\sigma = 0.0987921$	$\mu = 0.634195$, $\sigma = 0.0881635$
Straight_6	$\mu = 0.246331$, $\sigma = 0.247331$	$\mu = 0.414212$, $\sigma = 0.206362$	$\mu = 0.484263$, $\sigma = 0.163736$	$\mu = 0.537104$, $\sigma = 0.129482$	$\mu = 0.551841$, $\sigma = 0.120289$	$\mu = 0.593134$, $\sigma = 0.0980055$	$\mu = 0.633252$, $\sigma = 0.0865945$
Straight_7	$\mu = 0.246738$, $\sigma = 0.247718$	$\mu = 0.414546$, $\sigma = 0.206625$	$\mu = 0.484835$, $\sigma = 0.163954$	$\mu = 0.538068$, $\sigma = 0.129658$	$\mu = 0.552814$, $\sigma = 0.120656$	$\mu = 0.595474$, $\sigma = 0.0985157$	$\mu = 0.637484$, $\sigma = 0.0879285$
Straight_8	$\mu = 0.247998$, $\sigma = 0.248991$	$\mu = 0.416313$, $\sigma = 0.207556$	$\mu = 0.487358$, $\sigma = 0.164812$	$\mu = 0.541215$, $\sigma = 0.130701$	$\mu = 0.557048$, $\sigma = 0.121594$	$\mu = 0.600892$, $\sigma = 0.0995365$	$\mu = 0.647816$, $\sigma = 0.090552$
Woodbox	$\mu = 0.23858$, $\sigma = 0.238404$	$\mu = 0.41305$, $\sigma = 0.207586$	$\mu = 0.485021$, $\sigma = 0.159894$	$\mu = 0.528727$, $\sigma = 0.122419$	$\mu = 0.540441$, $\sigma = 0.111702$	$\mu = 0.572171$, $\sigma = 0.0865143$	$\mu = 0.601087$, $\sigma = 0.0694052$

Table E.4: Summary of the results from the Linear model (Metric=SIFT)

Linear results (SIFT)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
Chessboard	DNF	DNF	DNF	$\mu = 164.686$, $\sigma = 69.9408$	DNF	DNF	DNF
Circle_3	$\mu = 24.8116$, $\sigma = 27.7754$	$\mu = 40.8326$, $\sigma = 26.4081$	$\mu = 59.0161$, $\sigma = 28.8146$	$\mu = 75.1722$, $\sigma = 30.204$	$\mu = 81.9541$, $\sigma = 30.412$	$\mu = 89.9445$, $\sigma = 29.1429$	$\mu = 94.097$, $\sigma = 24.2138$
Circle_4	$\mu = 26.6764$, $\sigma = 29.6303$	$\mu = 42.7378$, $\sigma = 26.7155$	$\mu = 62.8255$, $\sigma = 30.3848$	$\mu = 80.6427$, $\sigma = 32.0328$	$\mu = 87.592$, $\sigma = 30.7809$	$\mu = 96.4703$, $\sigma = 28.651$	$\mu = 103.934$, $\sigma = 27.1253$
Circle_5	$\mu = 26.1531$, $\sigma = 29.3785$	$\mu = 40.5775$, $\sigma = 26.281$	$\mu = 55.1937$, $\sigma = 25.59$	$\mu = 72.5487$, $\sigma = 28.1019$	$\mu = 74.6665$, $\sigma = 27.0652$	$\mu = 83.6158$, $\sigma = 24.77$	$\mu = 89.3866$, $\sigma = 26.9518$
Faces	$\mu = 71.8329$, $\sigma = 54.2882$	$\mu = 96.6114$, $\sigma = 43.3831$	DNF	DNF	DNF	DNF	DNF
Idris_circle	$\mu = 15.7525$, $\sigma = 19.4685$	$\mu = 32.1805$, $\sigma = 22.9295$	$\mu = 42.4775$, $\sigma = 24.4187$	$\mu = 44.1682$, $\sigma = 22.8042$	$\mu = 47.4332$, $\sigma = 23.3881$	$\mu = 59.464$, $\sigma = 25.2921$	$\mu = 66.3984$, $\sigma = 26.1608$
Idris_fig8	$\mu = 16.1457$, $\sigma = 20.5197$	$\mu = 33.3776$, $\sigma = 25.9364$	$\mu = 40.7437$, $\sigma = 25.2707$	$\mu = 45.7657$, $\sigma = 24.745$	$\mu = 49.7684$, $\sigma = 25.2401$	$\mu = 59.5946$, $\sigma = 26.91$	$\mu = 68.2963$, $\sigma = 29.312$
Idris_straight	$\mu = 13.2683$, $\sigma = 17.0848$	$\mu = 22.0345$, $\sigma = 17.8923$	$\mu = 26.9107$, $\sigma = 18.5583$	$\mu = 33.339$, $\sigma = 19.8949$	$\mu = 32.6157$, $\sigma = 19.9257$	$\mu = 36.7009$, $\sigma = 23.0407$	$\mu = 42.0272$, $\sigma = 22.1438$
Straight_1	$\mu = 12.6509$, $\sigma = 15.97$	$\mu = 27.0872$, $\sigma = 19.6341$	$\mu = 42.8482$, $\sigma = 23.4379$	$\mu = 54.8494$, $\sigma = 25.86$	$\mu = 52.2853$, $\sigma = 23.6144$	$\mu = 78.3381$, $\sigma = 29.2516$	$\mu = 86.3878$, $\sigma = 31.7933$
Straight_2	$\mu = 11.173$, $\sigma = 14.6412$	$\mu = 27.8083$, $\sigma = 20.1063$	$\mu = 37.6469$, $\sigma = 21.2178$	$\mu = 47.6026$, $\sigma = 24.9773$	$\mu = 52.8794$, $\sigma = 27.5643$	$\mu = 53.4894$, $\sigma = 20.6701$	$\mu = 74.4239$, $\sigma = 27.2678$
Straight_3	$\mu = 12.6991$, $\sigma = 15.7797$	$\mu = 27.4566$, $\sigma = 19.2735$	$\mu = 36.2758$, $\sigma = 21.7272$	$\mu = 43.2625$, $\sigma = 20.8708$	$\mu = 46.6562$, $\sigma = 19.8493$	$\mu = 57.2892$, $\sigma = 22.3234$	$\mu = 71.4867$, $\sigma = 26.0762$
Straight_4	$\mu = 12.367$, $\sigma = 15.4534$	$\mu = 25.7111$, $\sigma = 19.0236$	$\mu = 34.2539$, $\sigma = 19.3909$	$\mu = 43.0369$, $\sigma = 19.9461$	$\mu = 46.2261$, $\sigma = 21.3034$	$\mu = 59.2597$, $\sigma = 25.5825$	$\mu = 82.129$, $\sigma = 30.18$
Straight_5	$\mu = 12.3674$, $\sigma = 15.7667$	$\mu = 24.5058$, $\sigma = 18.5724$	$\mu = 34.4521$, $\sigma = 19.8479$	$\mu = 41.9142$, $\sigma = 19.6729$	$\mu = 41.4198$, $\sigma = 19.0895$	$\mu = 54.5975$, $\sigma = 21.6964$	$\mu = 76.0033$, $\sigma = 33.2191$
Straight_6	$\mu = 12.6146$, $\sigma = 15.5363$	$\mu = 25.7522$, $\sigma = 18.4387$	$\mu = 35.7371$, $\sigma = 19.2057$	$\mu = 41.8999$, $\sigma = 19.1119$	$\mu = 45.9051$, $\sigma = 18.764$	$\mu = 51.5305$, $\sigma = 18.9668$	$\mu = 67.8026$, $\sigma = 25.6017$
Straight_7	$\mu = 13.6907$, $\sigma = 17.2453$	$\mu = 27.4876$, $\sigma = 18.7034$	$\mu = 36.4941$, $\sigma = 19.9877$	$\mu = 44.1026$, $\sigma = 20.6967$	$\mu = 44.6928$, $\sigma = 19.6673$	$\mu = 55.9552$, $\sigma = 20.5227$	$\mu = 74.8307$, $\sigma = 28.5999$

Continued on next page...

Table E.4 – Continued

Linear results (SIFT)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
Straight_8	$\mu = 13.5753$, $\sigma = 17.7665$	$\mu = 25.6432$, $\sigma = 18.8713$	$\mu = 34.1584$, $\sigma = 20.3496$	$\mu = 44.3375$, $\sigma = 22.1407$	$\mu = 41.4261$, $\sigma = 18.4752$	$\mu = 55.8554$, $\sigma = 26.4712$	$\mu = 82.9314$, $\sigma = 33.339$
Woodbox	$\mu = 15.8906$, $\sigma = 22.3954$	$\mu = 28.1234$, $\sigma = 24.593$	$\mu = 35.3217$, $\sigma = 26.0775$	$\mu = 42.5416$, $\sigma = 27.5697$	$\mu = 44.9551$, $\sigma = 27.5676$	$\mu = 42.6688$, $\sigma = 26.6505$	DNF

Table E.5: Summary of the results from the Linear model (Metric=Euclidean)

Linear results (Euclidean)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DTEAPOT	$\mu = 670.967$, $\sigma = 410.838$	$\mu = 1180.82$, $\sigma = 329.807$	$\mu = 1543.9$, $\sigma = 320.159$	$\mu = 1924.48$, $\sigma = 343.579$	$\mu = 2075.91$, $\sigma = 349.179$	$\mu = 2445.44$, $\sigma = 437.818$	DNF
2DWOODBOX	$\mu = 799.188$, $\sigma = 599.916$	$\mu = 1327.26$, $\sigma = 677.559$	$\mu = 1676.35$, $\sigma = 780.68$	$\mu = 1983.84$, $\sigma = 866.866$	$\mu = 1980.89$, $\sigma = 816.884$	$\mu = 1901.28$, $\sigma = 560.724$	DNF
BMNOISE	$\mu = 1846.07$, $\sigma = 1946.18$	$\mu = 3035.83$, $\sigma = 1754.71$	$\mu = 3337.35$, $\sigma = 1577.61$	DNF	DNF	DNF	DNF
BRUSH	$\mu = 145.663$, $\sigma = 158.494$	$\mu = 335.499$, $\sigma = 206.93$	$\mu = 528.739$, $\sigma = 259.624$	$\mu = 838.212$, $\sigma = 339.467$	$\mu = 940.49$, $\sigma = 391.316$	$\mu = 1505.09$, $\sigma = 530.342$	$\mu = 2226.62$, $\sigma = 812.189$
CHESSBOARD	$\mu = 4057.39$, $\sigma = 2500.94$	$\mu = 7731.14$, $\sigma = 2037.39$	$\mu = 10183.1$, $\sigma = 1663.55$	$\mu = 15286.6$, $\sigma = 3165.05$	$\mu = 13131.6$, $\sigma = 2597.76$	DNF	DNF
CIRCLE_3	$\mu = 417.856$, $\sigma = 419.675$	$\mu = 909.752$, $\sigma = 476.611$	$\mu = 1293.77$, $\sigma = 503.145$	$\mu = 1683.89$, $\sigma = 535.878$	$\mu = 1806.95$, $\sigma = 539.605$	$\mu = 2139.12$, $\sigma = 502.017$	$\mu = 2347.52$, $\sigma = 416.214$
CIRCLE_4	$\mu = 425.679$, $\sigma = 427.664$	$\mu = 936.72$, $\sigma = 490.961$	$\mu = 1330.82$, $\sigma = 515.315$	$\mu = 1747.34$, $\sigma = 552.26$	$\mu = 1866.63$, $\sigma = 557.218$	$\mu = 2164.59$, $\sigma = 498.202$	$\mu = 2409.53$, $\sigma = 418.353$
CIRCLE_5	$\mu = 456.52$, $\sigma = 459.394$	$\mu = 987.248$, $\sigma = 513.892$	$\mu = 1365.67$, $\sigma = 519.908$	$\mu = 1702.14$, $\sigma = 505.555$	$\mu = 1797.03$, $\sigma = 492.476$	$\mu = 2070.37$, $\sigma = 456.759$	$\mu = 2219.42$, $\sigma = 378.757$
EXPT03	$\mu = 696.892$, $\sigma = 533.271$	$\mu = 1012.89$, $\sigma = 485.693$	DNF	DNF	DNF	DNF	DNF
FACES	$\mu = 2482.38$, $\sigma = 1726.55$	$\mu = 3546.16$, $\sigma = 1182.19$	DNF	DNF	DNF	DNF	DNF
IDRIS_CIRCLE	$\mu = 219.638$, $\sigma = 231.16$	$\mu = 558.744$, $\sigma = 324.725$	$\mu = 884.36$, $\sigma = 394.781$	$\mu = 1270.72$, $\sigma = 467.595$	$\mu = 1411.02$, $\sigma = 489.055$	$\mu = 1877.18$, $\sigma = 556.391$	$\mu = 2357.42$, $\sigma = 600.545$
IDRIS_FIG8	$\mu = 213.674$, $\sigma = 230.724$	$\mu = 551.628$, $\sigma = 353.182$	$\mu = 853.554$, $\sigma = 446.691$	$\mu = 1208.83$, $\sigma = 552.223$	$\mu = 1336.53$, $\sigma = 590.557$	$\mu = 1747.48$, $\sigma = 687.546$	$\mu = 2244.87$, $\sigma = 777.594$
IDRIS_STRAIGHT	$\mu = 124.38$, $\sigma = 125.206$	$\mu = 245.049$, $\sigma = 153.319$	$\mu = 322.152$, $\sigma = 178.859$	$\mu = 426.305$, $\sigma = 225.973$	$\mu = 467.126$, $\sigma = 258.543$	$\mu = 625.026$, $\sigma = 332.187$	$\mu = 811.137$, $\sigma = 402.981$
KNIGHT_FIGHTING	$\mu = 396.754$, $\sigma = 330.679$	$\mu = 579.215$, $\sigma = 277.207$	DNF	DNF	DNF	DNF	DNF
KNIGHT_KNEELING	$\mu = 386.898$, $\sigma = 295.899$	$\mu = 520.84$, $\sigma = 203.409$	DNF	DNF	DNF	DNF	DNF

Continued on next page...

Table E.5 – Continued

Linear results (Euclidean)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
KNIGHT_STANDING	$\mu = 432.581$, $\sigma = 362.71$	$\mu = 560.205$, $\sigma = 282.447$	DNF	DNF	DNF	DNF	DNF
SINECOS	$\mu = 15.3497$, $\sigma = 52.7286$	$\mu = 54.364$, $\sigma = 80.6879$	$\mu = 113.908$, $\sigma = 68.1696$	$\mu = 171.915$, $\sigma = 91.878$	$\mu = 187.81$, $\sigma = 91.9732$	DNF	DNF
STRAIGHT_1	$\mu = 177.889$, $\sigma = 181.004$	$\mu = 399.028$, $\sigma = 224.142$	$\mu = 609.458$, $\sigma = 247.795$	$\mu = 893.809$, $\sigma = 321.589$	$\mu = 1014.44$, $\sigma = 353.272$	$\mu = 1448.39$, $\sigma = 462.966$	$\mu = 1890.48$, $\sigma = 548.239$
STRAIGHT_2	$\mu = 172.217$, $\sigma = 174.155$	$\mu = 378.549$, $\sigma = 199.308$	$\mu = 580.831$, $\sigma = 236.383$	$\mu = 829.626$, $\sigma = 283.236$	$\mu = 938.438$, $\sigma = 313.659$	$\mu = 1310.09$, $\sigma = 384.804$	$\mu = 1711.5$, $\sigma = 477.661$
STRAIGHT_3	$\mu = 156.976$, $\sigma = 158.389$	$\mu = 318.945$, $\sigma = 164.85$	$\mu = 460.437$, $\sigma = 176.911$	$\mu = 650.86$, $\sigma = 215.699$	$\mu = 728.729$, $\sigma = 228.534$	$\mu = 950.913$, $\sigma = 267.969$	$\mu = 1266.8$, $\sigma = 321.991$
STRAIGHT_4	$\mu = 157.751$, $\sigma = 159.678$	$\mu = 322.464$, $\sigma = 168.935$	$\mu = 462.988$, $\sigma = 180.01$	$\mu = 659.011$, $\sigma = 215.872$	$\mu = 729.546$, $\sigma = 230.908$	$\mu = 983.097$, $\sigma = 275.708$	$\mu = 1293.2$, $\sigma = 340.575$
STRAIGHT_5	$\mu = 159.803$, $\sigma = 162.878$	$\mu = 329.948$, $\sigma = 178.475$	$\mu = 469.848$, $\sigma = 181.124$	$\mu = 664.641$, $\sigma = 222.16$	$\mu = 757.159$, $\sigma = 247.87$	$\mu = 1003.53$, $\sigma = 286.942$	$\mu = 1343.81$, $\sigma = 359.917$
STRAIGHT_6	$\mu = 160.417$, $\sigma = 162.129$	$\mu = 330.989$, $\sigma = 172.184$	$\mu = 479.116$, $\sigma = 183.106$	$\mu = 678.785$, $\sigma = 219.081$	$\mu = 762.92$, $\sigma = 236.209$	$\mu = 1014.9$, $\sigma = 280.978$	$\mu = 1336.03$, $\sigma = 340.756$
STRAIGHT_7	$\mu = 160.999$, $\sigma = 162.361$	$\mu = 329.414$, $\sigma = 169.681$	$\mu = 479.869$, $\sigma = 185.385$	$\mu = 686.521$, $\sigma = 225.36$	$\mu = 749.458$, $\sigma = 237.561$	$\mu = 1021.56$, $\sigma = 294.389$	$\mu = 1367.67$, $\sigma = 372.601$
STRAIGHT_8	$\mu = 161.952$, $\sigma = 163.884$	$\mu = 334.682$, $\sigma = 175.345$	$\mu = 488.403$, $\sigma = 193.802$	$\mu = 699.211$, $\sigma = 242.457$	$\mu = 789.957$, $\sigma = 261.529$	$\mu = 1067.71$, $\sigma = 326.871$	$\mu = 1473.64$, $\sigma = 435.28$
WOODBOX	$\mu = 277.445$, $\sigma = 285.213$	$\mu = 552.911$, $\sigma = 295.413$	$\mu = 719.143$, $\sigma = 273.534$	$\mu = 869.925$, $\sigma = 264.415$	$\mu = 922.233$, $\sigma = 265.388$	$\mu = 1138.26$, $\sigma = 324.025$	$\mu = 1443.67$, $\sigma = 449.659$

Appendix F

NURBS model

In this appendix we present our results from the NURBS model. In these tables, which are provided for each of the metrics we used to evaluate the results (Chapter 5), we state the average measured error between the ground truth and the output of the model, as well as the standard deviation of this error. Results are shown for all of the values of g (gap) we considered. A number of different configurations were considered — these results tables show the configuration which produced the lowest mean error. Configurations are listed along with their number in a table in Section F.1.

F.1 Configurations

Table F.1: Summary of the configurations for the NURBS model

NURBS configurations			
Configuration number	Degree	Control point strategy	Knot vector strategy
1	2	Global interpolate	Even
3	4	Global interpolate	Even
5	6	Global interpolate	Even
7	8	Global interpolate	Even
9	10	Global interpolate	Even
11	20	Global interpolate	Even
13	6	Global interpolate	Even
14	6	Global interpolate	Averages
15	6	Global interpolate	Interior multiples
19	8	Global interpolate	Even
20	8	Global interpolate	Averages

Continued on next page...

Table F.1 – Continued

NURBS configurations			
Configuration number	Degree	Control point strategy	Knot vector strategy
21	8	Global interpolate	Interior multiples
25	10	Global interpolate	Even
26	10	Global interpolate	Averages
27	10	Global interpolate	Interior multiples
31	20	Global interpolate	Even
32	20	Global interpolate	Averages
33	20	Global interpolate	Interior multiples
37	8	Global interpolate	Even
38	8	Global interpolate	Averages
39	8	Global interpolate	Interior multiples
43	8	Global approximate	Even
44	8	Global approximate	Averages
45	8	Global approximate	Interior multiples
49	10	Global interpolate	Even
50	10	Global interpolate	Averages
51	10	Global interpolate	Interior multiples
55	10	Global approximate	Even
56	10	Global approximate	Averages
57	10	Global approximate	Interior multiples
61	20	Global interpolate	Even
62	20	Global interpolate	Averages
63	20	Global interpolate	Interior multiples
67	20	Global approximate	Even
68	20	Global approximate	Averages
69	20	Global approximate	Interior multiples

F.2 Summary tables

Table F.2: Summary of the results from the NURBS model (Metric=taxi)

NURBS results (taxi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2dteapot	$\mu = 23171.5$, $\sigma = 15638.7$, 1	$\mu = 41254.6$, $\sigma = 14087.4$	$\mu = 55886.4$, $\sigma = 16016.8$, 1	$\mu = 71128$, $\sigma = 19574.5$, 1	$\mu = 77118.6$, $\sigma = 21204$, 1	$\mu = 128473$, $\sigma = 34369.6$	DNF
2dwoodbox	$\mu = 54685.3$, $\sigma = 46647.6$	$\mu = 87247.7$, $\sigma = 55840$, 1	$\mu = 107246$, $\sigma = 66025.2$	$\mu = 124560$, $\sigma = 71215.6$	$\mu = 117697$, $\sigma = 60617.8$	$\mu = 387223$, $\sigma = 176158$	DNF
Bmnoise	$\mu = 101260$, $\sigma = 106265$	$\mu = 152086$, $\sigma = 85306.6$	$\mu = 909427$, $\sigma = 456640$	DNF	DNF	DNF	DNF
Brush	$\mu = 11001.7$, $\sigma = 8689.87$, 1	$\mu = 17507.6$, $\sigma = 9060.61$, 1	$\mu = 23105.6$, $\sigma = 9692.28$, 1	$\mu = 33001.7$, $\sigma = 12526.7$, 1	$\mu = 36279$, $\sigma = 12610.6$, 1	$\mu = 53759.9$, $\sigma = 18686.5$, 1	$\mu = 316642$, $\sigma = 222431$, 1
Chessboard	$\mu = 214714$, $\sigma = 147704$	$\mu = 475319$, $\sigma = 164705$	$\mu = 823614$, $\sigma = 222207$	$\mu = 955686$, $\sigma = 340256$	$\mu = 955635$, $\sigma = 137098$	DNF	DNF
Circle_3	$\mu = 29484.3$, $\sigma = 29281.8$, 1	$\mu = 55364.8$, $\sigma = 32496.3$, 1	$\mu = 76028.1$, $\sigma = 35220.2$, 1	$\mu = 97617.4$, $\sigma = 37896.1$, 1	$\mu = 106892$, $\sigma = 39968.8$, 1	$\mu = 125239$, $\sigma = 36674.1$, 1	$\mu = 139684$, $\sigma = 30246.1$, 1
Circle_4	$\mu = 29111.7$, $\sigma = 29677.8$, 1	$\mu = 56263.6$, $\sigma = 32936$, 1	$\mu = 77263.2$, $\sigma = 35517.1$, 1	$\mu = 101056$, $\sigma = 38333.3$, 1	$\mu = 110248$, $\sigma = 40889.1$, 1	$\mu = 126358$, $\sigma = 36060.6$, 1	$\mu = 141672$, $\sigma = 29624.7$, 1
Circle_5	$\mu = 30475.3$, $\sigma = 31031.1$	$\mu = 58314.5$, $\sigma = 32834.1$	$\mu = 78862.1$, $\sigma = 34751.9$	$\mu = 97248.1$, $\sigma = 35038.1$	$\mu = 104355$, $\sigma = 35271$	$\mu = 119050$, $\sigma = 32163.5$	$\mu = 128292$, $\sigma = 26198.2$
Expt03	$\mu = 48594.7$, $\sigma = 36663.7$	$\mu = 192103$, $\sigma = 69369$	DNF	DNF	DNF	DNF	DNF
Faces	$\mu = 194710$, $\sigma = 141586$	$\mu = 453088$, $\sigma = 264821$	DNF	DNF	DNF	DNF	DNF
Idris_circle	$\mu = 14974.8$, $\sigma = 13406.1$, 1	$\mu = 30868.5$, $\sigma = 16422.4$, 1	$\mu = 45348.3$, $\sigma = 19441.4$, 1	$\mu = 62755.7$, $\sigma = 23760.7$, 1	$\mu = 70195.5$, $\sigma = 25847.5$, 1	$\mu = 92700.1$, $\sigma = 30837.2$, 1	$\mu = 117920$, $\sigma = 35949$, 1

Continued on next page...

Table F.2 – Continued

NURBS results (taxi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
IDRIS_FIG8	$\mu = 15298.6$, $\sigma = 13896.4$, 1	$\mu = 31358$, $\sigma = 18683.9$, 1	$\mu = 45600.1$, $\sigma = 23186.7$, 1	$\mu = 64121.2$, $\sigma = 30277.5$, 1	$\mu = 69838.6$, $\sigma = 31654.2$, 1	$\mu = 91741.2$, $\sigma = 36906.6$, 1	$\mu = 115681$, $\sigma = 41747.7$, 1
IDRIS_STRAIGHT	$\mu = 11292.3$, $\sigma = 8173.43$, 1	$\mu = 17177.5$, $\sigma = 8035.93$, 1	$\mu = 20739.8$, $\sigma = 8731.76$, 1	$\mu = 26218.8$, $\sigma = 10222.1$, 1	$\mu = 28496$, $\sigma = 11498.5$, 1	$\mu = 37849.4$, $\sigma = 14693.2$, 1	$\mu = 45175.4$, $\sigma = 16211.3$, 1
KNIGHT_FIGHTING	$\mu = 11002.8$, $\sigma = 8298.49$, 1	$\mu = 15462.1$, $\sigma = 6832.88$, 1	DNF	DNF	DNF	DNF	DNF
KNIGHT_KNEELING	$\mu = 14746.2$, $\sigma = 10824.5$, 1	$\mu = 20099.8$, $\sigma = 7704.5$, 1	DNF	DNF	DNF	DNF	DNF
KNIGHT_STANDING	$\mu = 18723.5$, $\sigma = 16054.1$, 1	$\mu = 21953.1$, $\sigma = 12456.9$, 3	DNF	DNF	DNF	DNF	DNF
SINECOS	$\mu = 45.3623$, $\sigma = 102.901$	$\mu = 136.46$, $\sigma = 175.652$, 1	$\mu = 259.835$, $\sigma = 152.063$	$\mu = 347.754$, $\sigma = 182.436$	$\mu = 547.216$, $\sigma = 254.043$	DNF	DNF
STRAIGHT_1	$\mu = 15627.2$, $\sigma = 12466.3$, 1	$\mu = 27115.1$, $\sigma = 13736.6$, 1	$\mu = 37347.6$, $\sigma = 14033.7$, 1	$\mu = 50958.6$, $\sigma = 17233.5$, 1	$\mu = 57498.9$, $\sigma = 20490.1$, 1	$\mu = 79344.1$, $\sigma = 26956.9$, 1	$\mu = 105250$, $\sigma = 34318.3$, 1
STRAIGHT_2	$\mu = 15001.5$, $\sigma = 12783$, 1	$\mu = 25882.1$, $\sigma = 12502.6$, 1	$\mu = 35343.6$, $\sigma = 13218.7$, 1	$\mu = 46514.2$, $\sigma = 14822.5$, 1	$\mu = 51500.2$, $\sigma = 16848.8$, 1	$\mu = 69270.5$, $\sigma = 21481.9$, 1	$\mu = 90039.6$, $\sigma = 26997.8$, 1
STRAIGHT_3	$\mu = 14147.3$, $\sigma = 12098.1$, 1	$\mu = 23201.9$, $\sigma = 10951.8$, 1	$\mu = 29926.7$, $\sigma = 11188.6$, 1	$\mu = 38360.6$, $\sigma = 11806.6$, 1	$\mu = 41892.4$, $\sigma = 12225.5$, 1	$\mu = 52281.9$, $\sigma = 14467.2$, 1	$\mu = 70503.7$, $\sigma = 19890.5$, 1
STRAIGHT_4	$\mu = 14183.3$, $\sigma = 12073.7$, 1	$\mu = 23304$, $\sigma = 10889.4$, 1	$\mu = 30183.8$, $\sigma = 11217.7$, 1	$\mu = 39095.3$, $\sigma = 12181.7$, 1	$\mu = 42002.5$, $\sigma = 12565.4$, 1	$\mu = 54200.4$, $\sigma = 15417.6$, 1	$\mu = 72317.1$, $\sigma = 21269.9$, 1

Continued on next page...

Table F.2 – Continued

NURBS results (taxi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
Straight_5	$\mu = 14302.6$, $\sigma = 12189.7$, 1	$\mu = 23659.5$, $\sigma = 11294.3$, 1	$\mu = 30524$, $\sigma = 11326$, 1	$\mu = 39854.7$, $\sigma = 12939$, 1	$\mu = 44080.8$, $\sigma = 13812.4$, 1	$\mu = 55700.3$, $\sigma = 16062.5$, 1	$\mu = 75600.7$, $\sigma = 22752.8$, 1
Straight_6	$\mu = 14503.8$, $\sigma = 12511.7$, 1	$\mu = 23852.5$, $\sigma = 11234.4$, 1	$\mu = 31293.8$, $\sigma = 11787.7$, 1	$\mu = 40453.7$, $\sigma = 12716.7$, 1	$\mu = 44529.3$, $\sigma = 13328.6$, 1	$\mu = 56410.8$, $\sigma = 16159.6$, 1	$\mu = 75354.6$, $\sigma = 21701.6$, 1
Straight_7	$\mu = 14373.3$, $\sigma = 12189.2$, 1	$\mu = 23602.6$, $\sigma = 10825.4$, 1	$\mu = 31026.7$, $\sigma = 11382.5$, 1	$\mu = 40906.1$, $\sigma = 12777.1$, 1	$\mu = 43607.4$, $\sigma = 13092.2$, 1	$\mu = 56901.8$, $\sigma = 16632.3$, 1	$\mu = 77240$, $\sigma = 23181.6$, 1
Straight_8	$\mu = 14361$, $\sigma = 12177.7$, 1	$\mu = 23706$, $\sigma = 10893.7$, 1	$\mu = 31247.9$, $\sigma = 11382.9$, 1	$\mu = 41204.7$, $\sigma = 12743.4$, 1	$\mu = 45504.3$, $\sigma = 13841.6$, 1	$\mu = 59975.3$, $\sigma = 18098.2$, 1	$\mu = 84715.8$, $\sigma = 27465.3$, 1
Woodbox	$\mu = 19462.3$, $\sigma = 19281.1$	$\mu = 33929.2$, $\sigma = 19944.9$	$\mu = 43334.6$, $\sigma = 19449.5$, 1	$\mu = 51991$, $\sigma = 20252.5$, 1	$\mu = 54904.6$, $\sigma = 21060$	$\mu = 66694$, $\sigma = 25831.5$	$\mu = 83698.9$, $\sigma = 33939.2$, 1

Table F.3: Summary of the results from the NURBS model (Metric=pdiff)

NURBS results (pdiff)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DTEAPOT	$\mu = 234.531$, $\sigma = 161.176$, 1	$\mu = 349.079$, $\sigma = 111.096$	$\mu = 387.848$, $\sigma = 89.7645$, 1	$\mu = 415.752$, $\sigma = 72.8876$, 1	$\mu = 421.482$, $\sigma = 77.2805$, 1	$\mu = 414.123$, $\sigma = 87.7837$	DNF
2DWOODBOX	$\mu = 753.365$, $\sigma = 516.786$	$\mu = 998.478$, $\sigma = 346.55$, 1	$\mu = 1079.04$, $\sigma = 276.067$	$\mu = 1192.92$, $\sigma = 251.492$	$\mu = 1193.89$, $\sigma = 295.021$	$\mu = 1647.53$, $\sigma = 596.929$	DNF
BMNOISE	$\mu = 1152.63$, $\sigma = 1215.07$	$\mu = 1834.66$, $\sigma = 1059.26$	$\mu = 2379.07$, $\sigma = 475.171$	DNF	DNF	DNF	DNF
BRUSH	$\mu = 105.892$, $\sigma = 138.818$, 1	$\mu = 159.074$, $\sigma = 112.383$, 1	$\mu = 190.176$, $\sigma = 92.4425$, 1	$\mu = 241.909$, $\sigma = 92.9616$, 1	$\mu = 244.524$, $\sigma = 93.4923$, 1	$\mu = 431.366$, $\sigma = 263.158$, 1	$\mu = 1775.78$, $\sigma = 731.674$, 1
CHESSBOARD	$\mu = 383.776$, $\sigma = 281.382$	$\mu = 826.747$, $\sigma = 296.275$	$\mu = 1418.52$, $\sigma = 517.452$	$\mu = 1249.26$, $\sigma = 446.097$	$\mu = 1346.89$, $\sigma = 201.966$	DNF	DNF
CIRCLE_3	$\mu = 767.074$, $\sigma = 782.829$, 1	$\mu = 1259.17$, $\sigma = 650.263$, 1	$\mu = 1466.8$, $\sigma = 516.928$, 1	$\mu = 1607.5$, $\sigma = 402.785$, 1	$\mu = 1649.82$, $\sigma = 373.153$, 1	$\mu = 1739.55$, $\sigma = 292.719$, 1	$\mu = 1804.21$, $\sigma = 220.294$, 1
CIRCLE_4	$\mu = 776.31$, $\sigma = 790.92$, 1	$\mu = 1276.47$, $\sigma = 656.539$, 1	$\mu = 1490.14$, $\sigma = 520.468$, 1	$\mu = 1635.92$, $\sigma = 404.754$, 1	$\mu = 1677.15$, $\sigma = 377.077$, 1	$\mu = 1762.06$, $\sigma = 294.374$, 1	$\mu = 1831.3$, $\sigma = 220.663$, 1
CIRCLE_5	$\mu = 790.311$, $\sigma = 804.547$	$\mu = 1290.78$, $\sigma = 664.054$	$\mu = 1491.14$, $\sigma = 520.814$	$\mu = 1624.87$, $\sigma = 401.28$	$\mu = 1657.04$, $\sigma = 369.098$	$\mu = 1741.7$, $\sigma = 285.586$	$\mu = 1795.6$, $\sigma = 213.933$
EXPT03	$\mu = 333.105$, $\sigma = 527.296$	$\mu = 502.938$, $\sigma = 469.744$	DNF	DNF	DNF	DNF	DNF
FACES	$\mu = 483.165$, $\sigma = 411.167$	$\mu = 1142.86$, $\sigma = 589.358$	DNF	DNF	DNF	DNF	DNF
IDRIS_CIRCLE	$\mu = 271.719$, $\sigma = 320.36$, 1	$\mu = 436.802$, $\sigma = 269.409$, 1	$\mu = 528.343$, $\sigma = 233.681$, 1	$\mu = 631.201$, $\sigma = 211.808$, 1	$\mu = 674.598$, $\sigma = 212.824$, 1	$\mu = 797.791$, $\sigma = 209.924$, 1	$\mu = 925.081$, $\sigma = 213.733$, 1

Continued on next page...

Table F.3 – Continued

NURBS results (pdiff)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
IDRIS_FIG8	$\mu = 265.049$, $\sigma = 331.119$, 1	$\mu = 429.776$, $\sigma = 289.893$, 1	$\mu = 526.511$, $\sigma = 262.975$, 1	$\mu = 640.625$, $\sigma = 262.9$, 1	$\mu = 674.189$, $\sigma = 260.69$, 1	$\mu = 798.51$, $\sigma = 258.82$, 1	$\mu = 925.99$, $\sigma = 256.738$, 1
IDRIS_STRAIGHT	$\mu = 215.851$, $\sigma = 264.431$, 1	$\mu = 333.529$, $\sigma = 213.166$, 1	$\mu = 377.374$, $\sigma = 174.641$, 1	$\mu = 415.764$, $\sigma = 142.986$, 1	$\mu = 435.938$, $\sigma = 137.425$, 1	$\mu = 516.357$, $\sigma = 163.547$, 1	$\mu = 553.989$, $\sigma = 145.055$, 1
KNIGHT_FIGHTING	$\mu = 32.0593$, $\sigma = 26.8142$, 1	$\mu = 62.5521$, $\sigma = 26.1306$, 1	DNF	DNF	DNF	DNF	DNF
KNIGHT_KNEELING	$\mu = 51.237$, $\sigma = 41.4075$, 1	$\mu = 100.458$, $\sigma = 38.139$, 1	DNF	DNF	DNF	DNF	DNF
KNIGHT_STANDING	$\mu = 75.6444$, $\sigma = 59.7087$, 1	$\mu = 134.229$, $\sigma = 45.9512$, 1	DNF	DNF	DNF	DNF	DNF
SINECOS	$\mu = 0.107872$, $\sigma = 0.31422$	$\mu = 0.244802$, $\sigma = 0.543562$, 7	$\mu = 0.883998$, $\sigma = 0.602582$	$\mu = 1.04819$, $\sigma = 0.601949$	$\mu = 1.35059$, $\sigma = 0.549219$	DNF	DNF
STRAIGHT_1	$\mu = 736.682$, $\sigma = 745.025$, 1	$\mu = 1169.48$, $\sigma = 596.764$, 1	$\mu = 1341.1$, $\sigma = 464.443$, 1	$\mu = 1452.39$, $\sigma = 357.292$, 1	$\mu = 1484.2$, $\sigma = 331.018$, 1	$\mu = 1585.99$, $\sigma = 273.995$, 1	$\mu = 1651.08$, $\sigma = 238.291$, 1
STRAIGHT_2	$\mu = 727.937$, $\sigma = 737.086$, 1	$\mu = 1154.31$, $\sigma = 589.665$, 1	$\mu = 1312.12$, $\sigma = 455.697$, 1	$\mu = 1421.45$, $\sigma = 351.164$, 1	$\mu = 1447.96$, $\sigma = 322.529$, 1	$\mu = 1525.63$, $\sigma = 263.243$, 1	$\mu = 1594.25$, $\sigma = 227.044$, 1
STRAIGHT_3	$\mu = 729.824$, $\sigma = 740.122$, 1	$\mu = 1149.29$, $\sigma = 588.381$, 1	$\mu = 1300.03$, $\sigma = 452.26$, 1	$\mu = 1398.22$, $\sigma = 344.991$, 1	$\mu = 1419.44$, $\sigma = 317.128$, 1	$\mu = 1488.47$, $\sigma = 249.777$, 1	$\mu = 1562.2$, $\sigma = 217.219$, 1
STRAIGHT_4	$\mu = 729.383$, $\sigma = 738.49$, 1	$\mu = 1150.43$, $\sigma = 587.265$, 1	$\mu = 1299.2$, $\sigma = 450.092$, 1	$\mu = 1391.79$, $\sigma = 343.345$, 1	$\mu = 1420.82$, $\sigma = 317.918$, 1	$\mu = 1495.77$, $\sigma = 252.52$, 1	$\mu = 1569.92$, $\sigma = 223.178$, 1

Continued on next page...

Table F.3 – Continued

NURBS results (pdiff)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
Straight_5	$\mu = 733.322$, $\sigma = 741.708$, 1	$\mu = 1152.31$, $\sigma = 587.859$, 1	$\mu = 1302.55$, $\sigma = 450.703$, 1	$\mu = 1402.21$, $\sigma = 345.705$, 1	$\mu = 1433.2$, $\sigma = 319.864$, 1	$\mu = 1497.83$, $\sigma = 252.471$, 1	$\mu = 1583.11$, $\sigma = 226.179$, 1
Straight_6	$\mu = 741.211$, $\sigma = 750.523$, 1	$\mu = 1168.72$, $\sigma = 596.463$, 1	$\mu = 1324.55$, $\sigma = 458.838$, 1	$\mu = 1417.83$, $\sigma = 348.772$, 1	$\mu = 1445.9$, $\sigma = 321.771$, 1	$\mu = 1512.88$, $\sigma = 254.395$, 1	$\mu = 1594.83$, $\sigma = 220.296$, 1
Straight_7	$\mu = 743.398$, $\sigma = 752.673$, 1	$\mu = 1170.85$, $\sigma = 597.531$, 1	$\mu = 1329.64$, $\sigma = 460.506$, 1	$\mu = 1431.82$, $\sigma = 352.726$, 1	$\mu = 1460.11$, $\sigma = 327.829$, 1	$\mu = 1545.85$, $\sigma = 266.998$, 1	$\mu = 1590.13$, $\sigma = 227.615$, 1
Straight_8	$\mu = 743.276$, $\sigma = 751.204$, 1	$\mu = 1168.98$, $\sigma = 595.696$, 1	$\mu = 1339.65$, $\sigma = 462.612$, 1	$\mu = 1452.31$, $\sigma = 357.728$, 1	$\mu = 1489.06$, $\sigma = 332.943$, 1	$\mu = 1590.12$, $\sigma = 273.431$, 1	$\mu = 1705.19$, $\sigma = 251.637$, 1
Woodbox	$\mu = 259.233$, $\sigma = 291.916$	$\mu = 478.306$, $\sigma = 294.2$	$\mu = 583.776$, $\sigma = 273.413$, 1	$\mu = 653.127$, $\sigma = 257.238$, 1	$\mu = 668.613$, $\sigma = 254.164$	$\mu = 715.869$, $\sigma = 249.706$	$\mu = 765.749$, $\sigma = 256.946$, 1

Table F.4: Summary of the results from the NURBS model (Metric=mi)

NURBS results (mi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2dteapot	$\mu = 0.259581$, $\sigma = 0.15244$, 1	$\mu = 0.382671$, $\sigma = 0.0848305$	$\mu = 0.447164$, $\sigma = 0.0616559$, 1	$\mu = 0.502732$, $\sigma = 0.0579188$, 1	$\mu = 0.52379$, $\sigma = 0.0574138$, 1	$\mu = 0.673184$, $\sigma = 0.0883648$	DNF
2dwoodbox	$\mu = 0.464817$, $\sigma = 0.27403$	$\mu = 0.613814$, $\sigma = 0.135287$, 1	$\mu = 0.649547$, $\sigma = 0.0807596$	$\mu = 0.670193$, $\sigma = 0.0512356$	$\mu = 0.673323$, $\sigma = 0.0504617$	$\mu = 0.739547$, $\sigma = 0.0726898$	DNF
Bmnoise	$\mu = 0.380372$, $\sigma = 0.380415$	$\mu = 0.642235$, $\sigma = 0.337114$	$\mu = 0.83635$, $\sigma = 0.243494$	DNF	DNF	DNF	DNF
Brush	$\mu = 0.188648$, $\sigma = 0.169716$, 1	$\mu = 0.297183$, $\sigma = 0.139761$, 1	$\mu = 0.337435$, $\sigma = 0.108696$, 1	$\mu = 0.375133$, $\sigma = 0.093041$, 1	$\mu = 0.378844$, $\sigma = 0.0782449$, 1	$\mu = 0.407819$, $\sigma = 0.0757511$, 1	$\mu = 0.443743$, $\sigma = 0.0749446$, 1
Chessboard	$\mu = 0.465383$, $\sigma = 0.26928$	$\mu = 0.719311$, $\sigma = 0.182278$	$\mu = 0.897833$, $\sigma = 0.17329$	$\mu = 0.89778$, $\sigma = 0.17326$	$\mu = 0.788021$, $\sigma = 0.126117$	DNF	DNF
Circle_3	$\mu = 0.309929$, $\sigma = 0.282082$, 1	$\mu = 0.496364$, $\sigma = 0.232674$, 1	$\mu = 0.569526$, $\sigma = 0.182169$, 1	$\mu = 0.61733$, $\sigma = 0.139066$, 1	$\mu = 0.630464$, $\sigma = 0.127338$, 1	$\mu = 0.659488$, $\sigma = 0.0956542$, 1	$\mu = 0.680059$, $\sigma = 0.0691286$, 3
Circle_4	$\mu = 0.316033$, $\sigma = 0.27751$, 1	$\mu = 0.498107$, $\sigma = 0.23205$, 1	$\mu = 0.571319$, $\sigma = 0.182042$, 1	$\mu = 0.620085$, $\sigma = 0.137992$, 1	$\mu = 0.633709$, $\sigma = 0.127719$, 1	$\mu = 0.661581$, $\sigma = 0.0988652$, 1	$\mu = 0.680182$, $\sigma = 0.0688339$, 3
Circle_5	$\mu = 0.317055$, $\sigma = 0.282486$	$\mu = 0.501864$, $\sigma = 0.233928$	$\mu = 0.573818$, $\sigma = 0.184789$	$\mu = 0.621445$, $\sigma = 0.139815$	$\mu = 0.634048$, $\sigma = 0.127303$	$\mu = 0.661463$, $\sigma = 0.0987731$	$\mu = 0.678994$, $\sigma = 0.0716038$
Expt03	$\mu = 0.654401$, $\sigma = 0.2858$	$\mu = 0.794193$, $\sigma = 0.121783$	DNF	DNF	DNF	DNF	DNF
Faces	$\mu = 0.390466$, $\sigma = 0.216219$	$\mu = 0.6282$, $\sigma = 0.202183$	DNF	DNF	DNF	DNF	DNF
Idris_circle	$\mu = 0.250765$, $\sigma = 0.213107$, 1	$\mu = 0.400043$, $\sigma = 0.179587$, 1	$\mu = 0.463893$, $\sigma = 0.144166$, 1	$\mu = 0.507367$, $\sigma = 0.112371$, 1	$\mu = 0.520101$, $\sigma = 0.101162$, 1	$\mu = 0.548925$, $\sigma = 0.0790268$, 1	$\mu = 0.570374$, $\sigma = 0.0625361$, 1

Continued on next page...

Table F.4 – Continued

NURBS results (mi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
IDRIS_FIG8	$\mu = 0.252461$, $\sigma = 0.212824$, 1	$\mu = 0.399209$, $\sigma = 0.182486$, 1	$\mu = 0.463342$, $\sigma = 0.144623$, 1	$\mu = 0.507817$, $\sigma = 0.111561$, 1	$\mu = 0.518585$, $\sigma = 0.101531$, 1	$\mu = 0.548034$, $\sigma = 0.0806796$, 1	$\mu = 0.570044$, $\sigma = 0.0634863$, 1
IDRIS_STRAIGHT	$\mu = 0.232152$, $\sigma = 0.202356$, 1	$\mu = 0.359733$, $\sigma = 0.164538$, 1	$\mu = 0.407746$, $\sigma = 0.130675$, 1	$\mu = 0.442165$, $\sigma = 0.10455$, 1	$\mu = 0.452403$, $\sigma = 0.0950975$, 1	$\mu = 0.480847$, $\sigma = 0.0780287$, 1	$\mu = 0.500946$, $\sigma = 0.0616861$, 1
KNIGHT_FIGHTING	$\mu = 0.409484$, $\sigma = 0.236574$, 1	$\mu = 0.654512$, $\sigma = 0.198484$, 1	DNF	DNF	DNF	DNF	DNF
KNIGHT_KNEELING	$\mu = 0.526568$, $\sigma = 0.317552$, 1	$\mu = 0.78055$, $\sigma = 0.184094$, 1	DNF	DNF	DNF	DNF	DNF
KNIGHT_STANDING	$\mu = 0.565271$, $\sigma = 0.348318$, 1	$\mu = 0.854567$, $\sigma = 0.18444$, 1	DNF	DNF	DNF	DNF	DNF
SINECOS	DNF	$\mu = 0.0137051$, $\sigma = 0.116264$, 1	$\mu = 0.00773349$, $\sigma = 0.0875996$	$\mu = 0.00832838$, $\sigma = 0.0908791$	$\mu = 0.0828402$, $\sigma = 0.275641$	DNF	DNF
STRAIGHT_1	$\mu = 0.275882$, $\sigma = 0.242943$, 1	$\mu = 0.436434$, $\sigma = 0.203151$, 1	$\mu = 0.506844$, $\sigma = 0.158255$, 1	$\mu = 0.554418$, $\sigma = 0.1256$, 1	$\mu = 0.567913$, $\sigma = 0.119092$, 1	$\mu = 0.605014$, $\sigma = 0.100255$, 1	$\mu = 0.634219$, $\sigma = 0.0866178$, 1
STRAIGHT_2	$\mu = 0.271912$, $\sigma = 0.24338$, 1	$\mu = 0.429662$, $\sigma = 0.204513$, 1	$\mu = 0.499857$, $\sigma = 0.156245$, 1	$\mu = 0.544526$, $\sigma = 0.122965$, 1	$\mu = 0.557099$, $\sigma = 0.11587$, 1	$\mu = 0.591517$, $\sigma = 0.0967665$, 1	$\mu = 0.619345$, $\sigma = 0.0827741$, 1
STRAIGHT_3	$\mu = 0.277574$, $\sigma = 0.248933$, 1	$\mu = 0.437056$, $\sigma = 0.203314$, 1	$\mu = 0.501003$, $\sigma = 0.164518$, 1	$\mu = 0.548168$, $\sigma = 0.123701$, 1	$\mu = 0.562067$, $\sigma = 0.11204$, 1	$\mu = 0.595347$, $\sigma = 0.0906895$, 1	$\mu = 0.631995$, $\sigma = 0.0869856$, 1
STRAIGHT_4	$\mu = 0.277704$, $\sigma = 0.249052$, 1	$\mu = 0.43741$, $\sigma = 0.203499$, 1	$\mu = 0.502139$, $\sigma = 0.164936$, 1	$\mu = 0.550083$, $\sigma = 0.124459$, 1	$\mu = 0.562377$, $\sigma = 0.112067$, 1	$\mu = 0.599228$, $\sigma = 0.0918507$, 1	$\mu = 0.633874$, $\sigma = 0.0874278$, 1

Continued on next page...

Table F.4 – Continued

NURBS results (mi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
Straight_5	$\mu = 0.278751$, $\sigma = 0.249587$, 1	$\mu = 0.439218$, $\sigma = 0.204303$, 1	$\mu = 0.503603$, $\sigma = 0.165223$, 1	$\mu = 0.552261$, $\sigma = 0.125122$, 1	$\mu = 0.568114$, $\sigma = 0.114087$, 1	$\mu = 0.602511$, $\sigma = 0.0926967$, 1	$\mu = 0.637094$, $\sigma = 0.0882777$, 1
Straight_6	$\mu = 0.278382$, $\sigma = 0.249351$, 1	$\mu = 0.438157$, $\sigma = 0.203775$, 1	$\mu = 0.504015$, $\sigma = 0.16544$, 1	$\mu = 0.551768$, $\sigma = 0.124567$, 1	$\mu = 0.566449$, $\sigma = 0.112976$, 1	$\mu = 0.599688$, $\sigma = 0.0916163$, 1	$\mu = 0.636082$, $\sigma = 0.0869218$, 1
Straight_7	$\mu = 0.278971$, $\sigma = 0.24997$, 1	$\mu = 0.438639$, $\sigma = 0.203885$, 1	$\mu = 0.50477$, $\sigma = 0.165379$, 1	$\mu = 0.554256$, $\sigma = 0.125096$, 1	$\mu = 0.565203$, $\sigma = 0.112179$, 1	$\mu = 0.601662$, $\sigma = 0.0918227$, 1	$\mu = 0.640798$, $\sigma = 0.0883361$, 1
Straight_8	$\mu = 0.280805$, $\sigma = 0.250968$, 1	$\mu = 0.441546$, $\sigma = 0.204593$, 1	$\mu = 0.508233$, $\sigma = 0.165998$, 1	$\mu = 0.558471$, $\sigma = 0.125913$, 1	$\mu = 0.572766$, $\sigma = 0.114276$, 1	$\mu = 0.610204$, $\sigma = 0.0936668$, 1	$\mu = 0.649162$, $\sigma = 0.0900837$, 1
Woodbox	$\mu = 0.285396$, $\sigma = 0.236588$	$\mu = 0.446115$, $\sigma = 0.201099$	$\mu = 0.511034$, $\sigma = 0.159255$, 1	$\mu = 0.551713$, $\sigma = 0.121479$, 1	$\mu = 0.562702$, $\sigma = 0.109157$	$\mu = 0.590008$, $\sigma = 0.086838$	$\mu = 0.614934$, $\sigma = 0.0691001$, 1

Table F.5: Summary of the results from the NURBS model (Metric=SIFT)

NURBS results (SIFT)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
Chessboard	DNF	DNF	DNF	$\mu = 164.686$, $\sigma = 69.9408$	DNF	DNF	DNF
Circle_3	$\mu = 24.2958$, $\sigma = 25.4679$, 1	$\mu = 44.4531$, $\sigma = 27.3637$, 1	$\mu = 60.8312$, $\sigma = 28.9762$, 1	$\mu = 75.2436$, $\sigma = 28.7918$, 1	$\mu = 81.6663$, $\sigma = 30.1991$, 1	$\mu = 90.494$, $\sigma = 29.1434$, 1	$\mu = 97.7444$, $\sigma = 26.9832$, 1
Circle_4	$\mu = 26.1455$, $\sigma = 26.608$, 1	$\mu = 47.371$, $\sigma = 29.1476$, 1	$\mu = 64.5692$, $\sigma = 29.1931$, 1	$\mu = 78.8831$, $\sigma = 29.5424$, 1	$\mu = 86.822$, $\sigma = 29.9812$, 1	$\mu = 95.1193$, $\sigma = 26.5565$, 1	$\mu = 100.803$, $\sigma = 25.801$, 1
Circle_5	$\mu = 25.3334$, $\sigma = 26.5167$	$\mu = 44.9422$, $\sigma = 27.8112$	$\mu = 61.4087$, $\sigma = 28.7644$	$\mu = 72.4996$, $\sigma = 27.4796$	$\mu = 77.08$, $\sigma = 27.3808$	$\mu = 86.9885$, $\sigma = 25.9995$	$\mu = 91.3825$, $\sigma = 28.6607$
Faces	$\mu = 90.6781$, $\sigma = 66.9283$	DNF	DNF	DNF	DNF	DNF	DNF
Idris_circle	$\mu = 18.5956$, $\sigma = 20.5981$, 1	$\mu = 33.9451$, $\sigma = 23.4938$, 1	$\mu = 42.5142$, $\sigma = 24.9514$, 1	$\mu = 48.3976$, $\sigma = 23.275$, 1	$\mu = 52.7829$, $\sigma = 25.1766$, 1	$\mu = 64.7666$, $\sigma = 28.4808$, 1	$\mu = 70.5171$, $\sigma = 28.1708$, 1
Idris_fig8	$\mu = 18.857$, $\sigma = 21.2903$, 1	$\mu = 35.3041$, $\sigma = 24.8803$, 1	$\mu = 41.9101$, $\sigma = 24.551$, 1	$\mu = 49.7144$, $\sigma = 25.1439$, 1	$\mu = 52.9939$, $\sigma = 25.5943$, 1	$\mu = 63.9245$, $\sigma = 28.6738$, 1	$\mu = 73.3501$, $\sigma = 31.9374$, 1
Idris_straight	$\mu = 14.3823$, $\sigma = 17.4746$, 1	$\mu = 24.1743$, $\sigma = 18.8874$, 1	$\mu = 27.6801$, $\sigma = 18.1898$, 1	$\mu = 34.9595$, $\sigma = 19.8737$, 1	$\mu = 34.2605$, $\sigma = 20.6082$, 1	$\mu = 37.8478$, $\sigma = 21.9981$, 1	$\mu = 43.548$, $\sigma = 23.2487$, 1
Straight_1	$\mu = 14.6457$, $\sigma = 16.7727$, 1	$\mu = 29.0668$, $\sigma = 19.6485$, 1	$\mu = 42.3923$, $\sigma = 23.9953$, 1	$\mu = 52.5309$, $\sigma = 25.6204$, 1	$\mu = 53.286$, $\sigma = 23.7642$, 1	$\mu = 74.0153$, $\sigma = 27.0069$, 1	$\mu = 86.2992$, $\sigma = 32.4716$, 1
Straight_2	$\mu = 14.1685$, $\sigma = 17.4043$, 1	$\mu = 30.18$, $\sigma = 21.6023$, 1	$\mu = 39.582$, $\sigma = 22.4401$, 1	$\mu = 44.5232$, $\sigma = 22.2992$, 1	$\mu = 55.2602$, $\sigma = 26.097$, 1	$\mu = 61.7807$, $\sigma = 28.0305$, 1	$\mu = 71.2619$, $\sigma = 24.8611$, 1
Straight_3	$\mu = 16.1839$, $\sigma = 16.5472$, 1	$\mu = 30.1946$, $\sigma = 19.9492$, 1	$\mu = 38.6087$, $\sigma = 21.5344$, 1	$\mu = 47.8923$, $\sigma = 22.3957$, 1	$\mu = 49.677$, $\sigma = 20.7205$, 1	$\mu = 65.3662$, $\sigma = 24.1168$, 1	$\mu = 70.3869$, $\sigma = 23.883$, 1

Continued on next page...

Table F.5 – Continued

NURBS results (SIFT)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
Straight_4	$\mu = 15.626$, $\sigma = 17.2681$, 1	$\mu = 28.5617$, $\sigma = 19.429$, 1	$\mu = 37.0946$, $\sigma = 19.8128$, 1	$\mu = 44.6927$, $\sigma = 19.7196$, 1	$\mu = 48.1284$, $\sigma = 20.5342$, 1	$\mu = 61.0729$, $\sigma = 23.0662$, 1	$\mu = 76.5035$, $\sigma = 26.6567$, 1
Straight_5	$\mu = 16.1118$, $\sigma = 17.663$, 1	$\mu = 28.1993$, $\sigma = 19.476$, 1	$\mu = 37.5367$, $\sigma = 21.144$, 1	$\mu = 41.099$, $\sigma = 18.4081$, 1	$\mu = 47.0091$, $\sigma = 20.5205$, 1	$\mu = 60.5773$, $\sigma = 24.0401$, 1	$\mu = 69.8262$, $\sigma = 28.5815$, 1
Straight_6	$\mu = 16.6329$, $\sigma = 17.7464$, 1	$\mu = 29.8094$, $\sigma = 19.982$, 1	$\mu = 36.5754$, $\sigma = 19.6696$, 1	$\mu = 43.3841$, $\sigma = 18.8524$, 1	$\mu = 50.4374$, $\sigma = 19.9345$, 1	$\mu = 57.4463$, $\sigma = 20.8643$, 1	$\mu = 70.3636$, $\sigma = 24.4701$, 1
Straight_7	$\mu = 17.504$, $\sigma = 18.8897$, 1	$\mu = 30.5996$, $\sigma = 20.4865$, 1	$\mu = 37.0472$, $\sigma = 20.2411$, 1	$\mu = 46.8763$, $\sigma = 20.2898$, 1	$\mu = 50.6749$, $\sigma = 21.8306$, 1	$\mu = 61.6443$, $\sigma = 22.821$, 1	$\mu = 80.8239$, $\sigma = 26.0091$, 1
Straight_8	$\mu = 16.2981$, $\sigma = 18.9613$, 1	$\mu = 28.4425$, $\sigma = 19.8699$, 1	$\mu = 36.1802$, $\sigma = 20.4902$, 1	$\mu = 48.0364$, $\sigma = 22.1652$, 1	$\mu = 50.5267$, $\sigma = 22.0311$, 1	$\mu = 57.3009$, $\sigma = 25.1087$, 1	$\mu = 79.4751$, $\sigma = 30.8131$, 1
Woodbox	DNF	DNF	$\mu = 33.7382$, $\sigma = 23.805$, 1	$\mu = 37.4295$, $\sigma = 23.9232$, 1	$\mu = 40.9503$, $\sigma = 24.8906$	$\mu = 44.4697$, $\sigma = 27.6602$	$\mu = 114.796$, $\sigma = 77.7547$, 3

Table F.6: Summary of the results from the NURBS model (Metric=Euclidean)

NURBS results (Euclidean)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DTEAPOT	$\mu = 805.253$, $\sigma = 509.208$, 1	$\mu = 1351.38$, $\sigma = 395.209$	$\mu = 1723.94$, $\sigma = 380.664$, 1	$\mu = 2051.11$, $\sigma = 378.335$, 1	$\mu = 2180.21$, $\sigma = 362.49$, 1	$\mu = 3181.94$, $\sigma = 783.203$	DNF
2DWOODBOX	$\mu = 1012.28$, $\sigma = 792.95$	$\mu = 1610.55$, $\sigma = 864.793$, 1	$\mu = 1946.54$, $\sigma = 948.685$	$\mu = 2224.93$, $\sigma = 982.829$	$\mu = 2142.93$, $\sigma = 837.774$	$\mu = 4961.43$, $\sigma = 2052.28$	DNF
BMNOISE	$\mu = 2085.18$, $\sigma = 2187.21$	$\mu = 3252$, $\sigma = 1847.79$	$\mu = 10905.6$, $\sigma = 5071.13$	DNF	DNF	DNF	DNF
BRUSH	$\mu = 213.232$, $\sigma = 186.241$, 1	$\mu = 396.13$, $\sigma = 222.314$, 1	$\mu = 594.788$, $\sigma = 282.628$, 1	$\mu = 900.137$, $\sigma = 362.788$, 1	$\mu = 1013.01$, $\sigma = 416.906$, 1	$\mu = 1565.84$, $\sigma = 533.87$, 1	$\mu = 5013.07$, $\sigma = 2806.82$, 1
CHESSBOARD	$\mu = 4397.77$, $\sigma = 2743.13$	$\mu = 7869.23$, $\sigma = 2096.36$	$\mu = 11209$, $\sigma = 2175.58$	$\mu = 15265.3$, $\sigma = 3158.58$	$\mu = 13570.7$, $\sigma = 1685.95$	DNF	DNF
CIRCLE_3	$\mu = 566.239$, $\sigma = 573.044$, 1	$\mu = 1112.47$, $\sigma = 636.585$, 1	$\mu = 1533.99$, $\sigma = 684.337$, 1	$\mu = 1941.45$, $\sigma = 705.143$, 1	$\mu = 2089.69$, $\sigma = 721.988$, 1	$\mu = 2364.58$, $\sigma = 620.544$, 1	$\mu = 2543.56$, $\sigma = 481.451$, 1
CIRCLE_4	$\mu = 569.224$, $\sigma = 581.231$, 1	$\mu = 1144.51$, $\sigma = 649.154$, 1	$\mu = 1569.67$, $\sigma = 683.876$, 1	$\mu = 2019.36$, $\sigma = 720.671$, 1	$\mu = 2153.28$, $\sigma = 730.41$, 1	$\mu = 2385.53$, $\sigma = 602.367$, 1	$\mu = 2598.65$, $\sigma = 475.811$, 1
CIRCLE_5	$\mu = 608.76$, $\sigma = 623.348$	$\mu = 1203.11$, $\sigma = 662.442$	$\mu = 1619.51$, $\sigma = 680.028$	$\mu = 1949.53$, $\sigma = 651.47$	$\mu = 2065.56$, $\sigma = 642.766$	$\mu = 2294.51$, $\sigma = 566.438$	$\mu = 2406.21$, $\sigma = 456.235$
EXPT03	$\mu = 788.382$, $\sigma = 575.429$	$\mu = 2468.96$, $\sigma = 719.503$	DNF	DNF	DNF	DNF	DNF
FACES	$\mu = 2872.92$, $\sigma = 2022.3$	$\mu = 6027.09$, $\sigma = 3100.55$	DNF	DNF	DNF	DNF	DNF
IDRIS_CIRCLE	$\mu = 281.751$, $\sigma = 267.819$, 1	$\mu = 683.93$, $\sigma = 380.507$, 1	$\mu = 1056.19$, $\sigma = 466.286$, 1	$\mu = 1464.23$, $\sigma = 551.531$, 1	$\mu = 1632.41$, $\sigma = 586.565$, 1	$\mu = 2119.42$, $\sigma = 655.634$, 1	$\mu = 2640.88$, $\sigma = 715.956$, 1

Continued on next page...

Table F.6 – Continued

NURBS results (Euclidean)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
IDRIS_FIG8	$\mu = 291.057$, $\sigma = 286.312$, 1	$\mu = 682.263$, $\sigma = 439.608$, 1	$\mu = 1031.36$, $\sigma = 554.669$, 1	$\mu = 1452.48$, $\sigma = 685.333$, 1	$\mu = 1581.05$, $\sigma = 712.158$, 1	$\mu = 2036.18$, $\sigma = 789.083$, 1	$\mu = 2516.55$, $\sigma = 837.668$, 1
IDRIS_STRAIGHT	$\mu = 175.885$, $\sigma = 134.372$, 1	$\mu = 289.897$, $\sigma = 153.574$, 1	$\mu = 367.002$, $\sigma = 182.08$, 1	$\mu = 485.424$, $\sigma = 237.387$, 1	$\mu = 535.686$, $\sigma = 271.441$, 1	$\mu = 723.364$, $\sigma = 336.999$, 1	$\mu = 902.946$, $\sigma = 402.963$, 1
KNIGHT_FIGHTING	$\mu = 446.6$, $\sigma = 355.55$, 1	$\mu = 683.593$, $\sigma = 311.678$, 1	DNF	DNF	DNF	DNF	DNF
KNIGHT_KNEELING	$\mu = 428.974$, $\sigma = 311.635$, 1	$\mu = 602.793$, $\sigma = 225.523$, 1	DNF	DNF	DNF	DNF	DNF
KNIGHT_STANDING	$\mu = 477.296$, $\sigma = 385.485$, 1	$\mu = 637.365$, $\sigma = 320.115$, 1	DNF	DNF	DNF	DNF	DNF
SINECOS	$\mu = 25.3397$, $\sigma = 57.4261$	$\mu = 71.6093$, $\sigma = 91.8039$, 1	$\mu = 119.747$, $\sigma = 70.9396$	$\mu = 160.537$, $\sigma = 79.7951$	$\mu = 254.311$, $\sigma = 112.73$	DNF	DNF
STRAIGHT_1	$\mu = 247.121$, $\sigma = 208.081$, 1	$\mu = 479.77$, $\sigma = 263.435$, 1	$\mu = 699.166$, $\sigma = 282.537$, 1	$\mu = 995.009$, $\sigma = 359.557$, 1	$\mu = 1137.32$, $\sigma = 424.073$, 1	$\mu = 1584.22$, $\sigma = 538.939$, 1	$\mu = 2035.28$, $\sigma = 628.023$, 1
STRAIGHT_2	$\mu = 238.45$, $\sigma = 211.678$, 1	$\mu = 453.666$, $\sigma = 232.834$, 1	$\mu = 668.904$, $\sigma = 273.538$, 1	$\mu = 928.232$, $\sigma = 325.436$, 1	$\mu = 1037.39$, $\sigma = 362.639$, 1	$\mu = 1417.6$, $\sigma = 440.732$, 1	$\mu = 1829.26$, $\sigma = 536.753$, 1
STRAIGHT_3	$\mu = 217.821$, $\sigma = 191.157$, 1	$\mu = 384.979$, $\sigma = 191.486$, 1	$\mu = 533.129$, $\sigma = 214.261$, 1	$\mu = 725.641$, $\sigma = 249.328$, 1	$\mu = 808.325$, $\sigma = 262.172$, 1	$\mu = 1019.27$, $\sigma = 295.996$, 1	$\mu = 1326.52$, $\sigma = 348.219$, 1
STRAIGHT_4	$\mu = 218.642$, $\sigma = 191.127$, 1	$\mu = 386.846$, $\sigma = 191.358$, 1	$\mu = 535.429$, $\sigma = 216.008$, 1	$\mu = 742.398$, $\sigma = 256.726$, 1	$\mu = 807.233$, $\sigma = 266.519$, 1	$\mu = 1054.63$, $\sigma = 307.363$, 1	$\mu = 1351.46$, $\sigma = 368.177$, 1

Continued on next page...

Table F.6 – Continued

NURBS results (Euclidean)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
Straight_5	$\mu = 220.569$, $\sigma = 192.372$, 1	$\mu = 393.969$, $\sigma = 200.318$, 1	$\mu = 543.858$, $\sigma = 217.559$, 1	$\mu = 754.58$, $\sigma = 269.445$, 1	$\mu = 841.705$, $\sigma = 286.306$, 1	$\mu = 1079.55$, $\sigma = 320.123$, 1	$\mu = 1409.02$, $\sigma = 394.224$, 1
Straight_6	$\mu = 222.913$, $\sigma = 196.902$, 1	$\mu = 396.293$, $\sigma = 196.771$, 1	$\mu = 556.272$, $\sigma = 223.794$, 1	$\mu = 761.779$, $\sigma = 260.635$, 1	$\mu = 853.645$, $\sigma = 279.204$, 1	$\mu = 1091.18$, $\sigma = 318.682$, 1	$\mu = 1400.32$, $\sigma = 373.543$, 1
Straight_7	$\mu = 221.649$, $\sigma = 191.468$, 1	$\mu = 393.342$, $\sigma = 188.579$, 1	$\mu = 554.832$, $\sigma = 217.98$, 1	$\mu = 777.472$, $\sigma = 266.452$, 1	$\mu = 836.32$, $\sigma = 278.52$, 1	$\mu = 1103.6$, $\sigma = 334.041$, 1	$\mu = 1432.11$, $\sigma = 402.671$, 1
Straight_8	$\mu = 222.749$, $\sigma = 193.208$, 1	$\mu = 398.509$, $\sigma = 192.168$, 1	$\mu = 564.544$, $\sigma = 221.834$, 1	$\mu = 784.46$, $\sigma = 266.965$, 1	$\mu = 879.918$, $\sigma = 293.054$, 1	$\mu = 1165.79$, $\sigma = 362.392$, 1	$\mu = 1559.46$, $\sigma = 474.72$, 1
Woodbox	$\mu = 390.762$, $\sigma = 392.35$	$\mu = 685.811$, $\sigma = 395.857$	$\mu = 859.496$, $\sigma = 375.658$, 1	$\mu = 1019.48$, $\sigma = 381.349$, 1	$\mu = 1073.44$, $\sigma = 387.297$	$\mu = 1302.44$, $\sigma = 474.316$	$\mu = 1633.88$, $\sigma = 623.7$, 1

Appendix G

PDE

In this appendix we present some of the numeric schemes we used which were too large to included in the main text. Additionally we present our results from the PDE model, using both solutions we discussed in Chapter 10. In these tables, which are provided for each of the metrics we used to evaluate the results (Chapter 5), we state the average measured error between the ground truth and the output of the model, as well as the standard deviation of this error. Results are shown for all of the values of g (gap) we considered, although the analytic PDE solution was only applicable to the 2-D case. A number of different configurations were considered — these results tables show the configuration which produced the lowest mean error. The configurations used were discussed in Chapter 10 and listed in the two tables in Section G.2.

In the end, because of long run times and generally similar results to the analytic solution, the numerical solver was only applied to a number of specific examples to demonstrate feasibility and provide results for the corresponding discussion.

G.1 Numerical Schemes for PDEs

Here we derive a numerical scheme for the 6^{th} 2-D PDE in the same way we did for the Laplacian in the text of Chapter 10. Multiplying out the 6^{th} order PDE of Equation 10.29 gives us:

$$\left(\frac{\partial^2}{\partial u^2}+\frac{\partial^2}{\partial v^2}\right)^3=\frac{\partial^6}{\partial u^6}+3\frac{\partial^6}{\partial u^4\partial v^2}+3\frac{\partial^6}{\partial u^2\partial v^4}+\frac{\partial^6}{\partial v^6} \tag{G.1}$$

We now use Equation 10.17 and Equation 10.25 to give:

$$\begin{aligned}\frac{\partial^6}{\partial u^6}\approx\ & \tfrac{1}{\Delta^6}\big[\ f(u+3\Delta,v)+f(u-3\Delta,v)-6f(u+2\Delta,v)-6f(u-2\Delta,v)\\ & +15f(u+\Delta,v)+15f(u-\Delta,v)-20f(u,v)\big]\end{aligned} \tag{G.2}$$

and similarly Equation 10.18 and Equation 10.26

$$\begin{aligned}\frac{\partial^6}{\partial v^6}\approx\ & \tfrac{1}{\Delta^6}\big[\ f(u,v+3\Delta)+f(u,v-3\Delta)-6f(u,v+2\Delta)-6f(u,v-2\Delta)\\ & +15f(u,v+\Delta)+15f(u,v-\Delta)-20f(u,v)\big]\end{aligned} \tag{G.3}$$

Equation 10.18 and Equation 10.25 give:

$$\begin{aligned}\frac{\partial^6}{\partial u^4\partial v^2}\approx\ & \tfrac{1}{\Delta^6}\big[\ f(u+2\Delta,v+\Delta)-2f(u+2\Delta,v)+f(u+2\Delta,v-\Delta)\\ & +f(u-2\Delta,v+\Delta)-2f(u-2\Delta,v)+f(u-2\Delta,v-\Delta)\\ & -4f(u+\Delta,v+\Delta)+8f(u+\Delta,v)-4f(u+\Delta,v-\Delta)\\ & -4f(u-\Delta,v+\Delta)+8f(u-\Delta,v)-4f(u-\Delta,v-\Delta)\\ & +6f(u,v+\Delta)-12f(u,v)+6f(u,v-\Delta)\big]\end{aligned} \tag{G.4}$$

and again Equation 10.17 and Equation 10.26 give:

$$\begin{aligned}\frac{\partial^6}{\partial u^2 \partial v^4} \approx \quad \tfrac{1}{\Delta^6}\big[\quad & f(u+\Delta, v+2\Delta) - 2f(u, v+2\Delta) + f(u-\Delta, v+2\Delta) \\ & + f(u+\Delta, v-2\Delta) - 2f(u, v-2\Delta) + f(u-\Delta, v-2\Delta) \\ & - 4f(u+\Delta, v+\Delta) + 8f(u, v+\Delta) - 4f(u-\Delta, v+\Delta) \\ & - 4f(u+\Delta, v-\Delta) + 8f(u, v-\Delta) - 4f(u-\Delta, v-\Delta) \\ & + 6f(u+\Delta, v) - 12f(u, v) + 6f(u-\Delta, v)\big] \end{aligned} \tag{G.5}$$

By substituting equations G.2, G.3, G.4 and G.5 into Equation G.1 we get the following finite difference scheme for our 6th order elliptic PDE:

$$\begin{aligned}\frac{1}{\Delta^6}\big[\qquad & f(u+3\Delta, v) + f(u-3\Delta, v) - 6f(u+2\Delta, v) \\ & - 6f(u-2\Delta, v) + 15f(u+\Delta, v) + 15f(u-\Delta, v) - 20f(u, v) \\ & + 3f(u+2\Delta, v+\Delta) - 6f(u+2\Delta, v) + 3f(u+2\Delta, v-\Delta) \\ & + 3f(u-2\Delta, v+\Delta) - 6f(u-2\Delta, v) + 3f(u-2\Delta, v-\Delta) \\ & - 12f(u+\Delta, v+\Delta) + 24f(u+\Delta, v) - 12f(u+\Delta, v-\Delta) \\ & - 12f(u-\Delta, v+\Delta) + 24f(u-\Delta, v) - 12f(u-\Delta, v-\Delta) \\ & + 18f(u, v+\Delta) - 36f(u, v) + 18f(u, v-\Delta) \\ & + 3f(u+\Delta, v+2\Delta) - 6f(u, v+2\Delta) + 3f(u-\Delta, v+2\Delta) \\ & + 3f(u+\Delta, v-2\Delta) - 6f(u, v-2\Delta) + 3f(u-\Delta, v-2\Delta) \\ & - 12f(u+\Delta, v+\Delta) + 24f(u, v+\Delta) - 12f(u-\Delta, v+\Delta) \\ & - 12f(u+\Delta, v-\Delta) + 24f(u, v-\Delta) - 12f(u-\Delta, v-\Delta) \\ & + 18f(u+\Delta, v) - 36f(u, v) + 18f(u-\Delta, v) \\ & + f(u, v+3\Delta) + f(u, v-3\Delta) - 6f(u, v+2\Delta) \\ & - 6f(u, v-2\Delta) + 15f(u, v+\Delta) + 15f(u, v-\Delta) - 20f(u, v)\big] \end{aligned} \tag{G.6}$$

G.2 Configurations

Table G.1: Summary of the configurations for the PDE model (analytic)

PDE configurations (analytic)				
Configuration number	Order	α	Fourier modes	Remainder on
1	2	0.0	1	Yes
2	2	0.0	1	No
3	2	0.0	2	Yes
4	2	0.0	2	No
5	2	0.0	6	Yes
6	2	0.0	6	No
7	2	0.0	10	Yes
8	2	0.0	10	No
9	2	0.0	20	Yes
10	2	0.0	20	No
11	2	0.0	50	Yes
12	2	0.0	50	No
13	2	0.5	1	Yes
14	2	0.5	1	No
15	2	0.5	2	Yes
16	2	0.5	2	No
17	2	0.5	6	Yes
18	2	0.5	6	No
19	2	0.5	10	Yes
20	2	0.5	10	No
21	2	0.5	20	Yes
22	2	0.5	20	No
23	2	0.5	50	Yes
24	2	0.5	50	No
25	2	1	1	Yes
26	2	1	1	No
27	2	1	2	Yes
28	2	1	2	No
29	2	1	6	Yes
30	2	1	6	No
31	2	1	10	Yes
32	2	1	10	No
33	2	1	20	Yes
34	2	1	20	No
35	2	1	50	Yes
36	2	1	50	No
37	2	2	1	Yes
38	2	2	1	No
39	2	2	2	Yes
40	2	2	2	No
41	2	2	6	Yes
42	2	2	6	No
43	2	2	10	Yes
44	2	2	10	No
45	2	2	20	Yes
46	2	2	20	No
47	2	2	50	Yes
48	2	2	50	No
49	2	10	1	Yes
50	2	10	1	No

Continued on next page...

Table G.1 – Continued

PDE configurations (analytic)				
Configuration number	Order	α	Fourier modes	Remainder on
51	2	10	2	Yes
52	2	10	2	No
53	2	10	6	Yes
54	2	10	6	No
55	2	10	10	Yes
56	2	10	10	No
57	2	10	20	Yes
58	2	10	20	No
59	2	10	50	Yes
60	2	10	50	No
61	2	-1	1	Yes
62	2	-1	1	No
63	2	-1	2	Yes
64	2	-1	2	No
65	2	-1	6	Yes
66	2	-1	6	No
67	2	-1	10	Yes
68	2	-1	10	No
69	2	-1	20	Yes
70	2	-1	20	No
71	2	-1	50	Yes
72	2	-1	50	No
73	4	0.0	1	Yes
74	4	0.0	1	No
75	4	0.0	2	Yes
76	4	0.0	2	No
77	4	0.0	6	Yes
78	4	0.0	6	No
79	4	0.0	10	Yes
80	4	0.0	10	No
81	4	0.0	20	Yes
82	4	0.0	20	No
83	4	0.0	50	Yes
84	4	0.0	50	No
85	4	0.5	1	Yes
86	4	0.5	1	No
87	4	0.5	2	Yes
88	4	0.5	2	No
89	4	0.5	6	Yes
90	4	0.5	6	No
91	4	0.5	10	Yes
92	4	0.5	10	No
93	4	0.5	20	Yes
94	4	0.5	20	No
95	4	0.5	50	Yes
96	4	0.5	50	No
97	4	1	1	Yes
98	4	1	1	No
99	4	1	2	Yes
100	4	1	2	No
101	4	1	6	Yes
102	4	1	6	No
103	4	1	10	Yes
104	4	1	10	No

Continued on next page...

Table G.1 – Continued

PDE configurations (analytic)				
Configuration number	Order	α	Fourier modes	Remainder on
105	4	1	20	Yes
106	4	1	20	No
107	4	1	50	Yes
108	4	1	50	No
109	4	2	1	Yes
110	4	2	1	No
111	4	2	2	Yes
112	4	2	2	No
113	4	2	6	Yes
114	4	2	6	No
115	4	2	10	Yes
116	4	2	10	No
117	4	2	20	Yes
118	4	2	20	No
119	4	2	50	Yes
120	4	2	50	No
121	4	10	1	Yes
122	4	10	1	No
123	4	10	2	Yes
124	4	10	2	No
125	4	10	6	Yes
126	4	10	6	No
127	4	10	10	Yes
128	4	10	10	No
129	4	10	20	Yes
130	4	10	20	No
131	4	10	50	Yes
132	4	10	50	No
133	4	-1	1	Yes
134	4	-1	1	No
135	4	-1	2	Yes
136	4	-1	2	No
137	4	-1	6	Yes
138	4	-1	6	No
139	4	-1	10	Yes
140	4	-1	10	No
141	4	-1	20	Yes
142	4	-1	20	No
143	4	-1	50	Yes
144	4	-1	50	No
145	6	0.0	1	Yes
146	6	0.0	1	No
147	6	0.0	2	Yes
148	6	0.0	2	No
149	6	0.0	6	Yes
150	6	0.0	6	No
151	6	0.0	10	Yes
152	6	0.0	10	No
153	6	0.0	20	Yes
154	6	0.0	20	No
155	6	0.0	50	Yes
156	6	0.0	50	No
157	6	0.5	1	Yes
158	6	0.5	1	No

Continued on next page...

Table G.1 – Continued

PDE configurations (analytic)				
Configuration number	Order	α	Fourier modes	Remainder on
159	6	0.5	2	Yes
160	6	0.5	2	No
161	6	0.5	6	Yes
162	6	0.5	6	No
163	6	0.5	10	Yes
164	6	0.5	10	No
165	6	0.5	20	Yes
166	6	0.5	20	No
167	6	0.5	50	Yes
168	6	0.5	50	No
169	6	1	1	Yes
170	6	1	1	No
171	6	1	2	Yes
172	6	1	2	No
173	6	1	6	Yes
174	6	1	6	No
175	6	1	10	Yes
176	6	1	10	No
177	6	1	20	Yes
178	6	1	20	No
179	6	1	50	Yes
180	6	1	50	No
181	6	2	1	Yes
182	6	2	1	No
183	6	2	2	Yes
184	6	2	2	No
185	6	2	6	Yes
186	6	2	6	No
187	6	2	10	Yes
188	6	2	10	No
189	6	2	20	Yes
190	6	2	20	No
191	6	2	50	Yes
192	6	2	50	No
193	6	10	1	Yes
194	6	10	1	No
195	6	10	2	Yes
196	6	10	2	No
197	6	10	6	Yes
198	6	10	6	No
199	6	10	10	Yes
200	6	10	10	No
201	6	10	20	Yes
202	6	10	20	No
203	6	10	50	Yes
204	6	10	50	No
205	6	-1	1	Yes
206	6	-1	1	No
207	6	-1	2	Yes
208	6	-1	2	No
209	6	-1	6	Yes
210	6	-1	6	No
211	6	-1	10	Yes
212	6	-1	10	No

Continued on next page...

Table G.1 – Continued

PDE configurations (analytic)				
Configuration number	Order	α	Fourier modes	Remainder on
213	6	-1	20	Yes
214	6	-1	20	No
215	6	-1	50	Yes
216	6	-1	50	0

Table G.2: Summary of the configurations for the PDE model (numeric)

PDE configurations (numeric)			
Configuration number	Order	Grid resolution	e
1	2	10	0.01
2	2	10	0.0001
3	2	10	0.1
4	2	20	0.01
5	2	20	0.0001
6	2	20	0.1
7	2	50	0.01
8	2	50	0.0001
9	2	50	0.1
10	4	10	0.01
11	4	10	0.0001
12	4	10	0.1
13	4	20	0.01
14	4	20	0.0001
15	4	20	0.1
16	4	50	0.01
17	4	50	0.0001
18	4	50	0.1
19	6	10	0.01
20	6	10	0.0001
21	6	10	0.1
22	6	20	0.01
23	6	20	0.0001
24	6	20	0.1
25	6	50	0.01
26	6	50	0.0001
27	6	50	0.1

G.3 Summary tables

G.3.1 Analytic

Table G.3: Summary of the results from the PDE model (Metric=taxi)

PDE results (taxi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DTEAPOT	$\mu = 29380.6$, $\sigma = 14294$, 29	$\mu = 43319.3$, $\sigma = 8647.9$, 30	$\mu = 55408.7$, $\sigma = 12370.5$, 30	$\mu = 73385.8$, $\sigma = 20325.5$, 30	DNF	$\mu = 104564$, $\sigma = 31304.3$, 30	DNF
2DWOODBOX	$\mu = 69225.3$, $\sigma = 47048.4$, 29	$\mu = 93730.9$, $\sigma = 50502.9$, 30	$\mu = 115036$, $\sigma = 66394.3$, 30	$\mu = 133149$, $\sigma = 47933.3$, 100	DNF	$\mu = 145150$, $\sigma = 58040.7$, 30	DNF
BMNOISE	$\mu = 140734$, $\sigma = 85645.8$, 99	$\mu = 200322$, $\sigma = 54618.8$, 100	$\mu = 202258$, $\sigma = 51180.8$, 30	DNF	DNF	DNF	DNF
KNIGHT_FIGHTING	$\mu = 16366.9$, $\sigma = 10321.2$, 30	$\mu = 17837.8$, $\sigma = 4059.44$, 101	DNF	DNF	DNF	DNF	DNF
KNIGHT_KNEELING	$\mu = 22094.9$, $\sigma = 13955.1$, 30	$\mu = 23637.2$, $\sigma = 6896.79$, 30	DNF	DNF	DNF	DNF	DNF
KNIGHT_STANDING	$\mu = 33026.7$, $\sigma = 29279.8$, 30	$\mu = 27088.5$, $\sigma = 11632.1$, 30	DNF	DNF	DNF	DNF	DNF
SINECOS	$\mu = 109.217$, $\sigma = 107.319$, 30	$\mu = 207.045$, $\sigma = 162.963$, 30	$\mu = 284.475$, $\sigma = 185.488$, 30	$\mu = 532.092$, $\sigma = 295.597$, 30	DNF	DNF	DNF

Table G.4: Summary of the results from the PDE model (Metric=pdiff)

PDE results (pdiff)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DTEAPOT	$\mu = 271.531$, $\sigma = 158.713$, 29	$\mu = 365.374$, $\sigma = 71.9237$, 29	$\mu = 385.317$, $\sigma = 70.657$, 30	$\mu = 406.432$, $\sigma = 71.5152$, 30	DNF	$\mu = 447.461$, $\sigma = 88.5836$, 30	DNF
2DWOODBOX	$\mu = 674.5$, $\sigma = 443.774$, 29	$\mu = 894.238$, $\sigma = 267.88$, 30	$\mu = 1026.48$, $\sigma = 277.663$, 30	$\mu = 1202.24$, $\sigma = 277.94$, 30	DNF	$\mu = 1657.38$, $\sigma = 244.126$, 30	DNF
BMNOISE	$\mu = 2238.5$, $\sigma = 221.833$, 99	$\mu = 2372.36$, $\sigma = 215.698$, 30	$\mu = 2414.61$, $\sigma = 109.101$, 30	DNF	DNF	DNF	DNF
KNIGHT_FIGHTING	$\mu = 48.0074$, $\sigma = 28.8764$, 30	$\mu = 61.7333$, $\sigma = 13.0791$, 101	DNF	DNF	DNF	DNF	DNF
KNIGHT_KNEELING	$\mu = 68.9111$, $\sigma = 40.0279$, 30	$\mu = 94.2292$, $\sigma = 26.0802$, 30	DNF	DNF	DNF	DNF	DNF
KNIGHT_STANDING	$\mu = 107.778$, $\sigma = 61.6918$, 30	$\mu = 136.733$, $\sigma = 35.4758$, 100	DNF	DNF	DNF	DNF	DNF
SINECOS	$\mu = 0.148887$, $\sigma = 0.370032$, 99	$\mu = 0.165217$, $\sigma = 0.437421$, 97	$\mu = 0.691463$, $\sigma = 0.686934$, 99	$\mu = 1.24807$, $\sigma = 0.636747$, 30	DNF	DNF	DNF

Table G.5: Summary of the results from the PDE model (Metric=mi)

PDE results (mi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DTEAPOT	$\mu = 0.379271$, $\sigma = 0.0615053$, 29	$\mu = 0.43167$, $\sigma = 0.0269704$, 30	$\mu = 0.457859$, $\sigma = 0.0344314$, 30	$\mu = 0.504171$, $\sigma = 0.0501824$, 30	DNF	$\mu = 0.594874$, $\sigma = 0.063967$, 30	DNF
2DWOODBOX	$\mu = 0.584326$, $\sigma = 0.107738$, 29	$\mu = 0.640996$, $\sigma = 0.0461136$, 97	$\mu = 0.664255$, $\sigma = 0.0477588$, 30	$\mu = 0.681103$, $\sigma = 0.0403364$, 30	DNF	$\mu = 0.694278$, $\sigma = 0.0204245$, 30	DNF
BMNOISE	$\mu = 0.776592$, $\sigma = 0.0640439$, 99	$\mu = 0.840473$, $\sigma = 0.0640487$, 100	$\mu = 0.844806$, $\sigma = 0.076929$, 30	DNF	DNF	DNF	DNF
KNIGHT_FIGHTING	$\mu = 0.515742$, $\sigma = 0.129862$, 30	$\mu = 0.605495$, $\sigma = 0.0466704$, 97	DNF	DNF	DNF	DNF	DNF
KNIGHT_KNEELING	$\mu = 0.678543$, $\sigma = 0.172645$, 30	$\mu = 0.769733$, $\sigma = 0.105676$, 30	DNF	DNF	DNF	DNF	DNF
KNIGHT_STANDING	$\mu = 0.738916$, $\sigma = 0.191864$, 30	$\mu = 0.856036$, $\sigma = 0.127705$, 30	DNF	DNF	DNF	DNF	DNF
SINECOS	$\mu = 0.00324675$, $\sigma = 0.0568877$, 100	$\mu = 0.00803403$, $\sigma = 0.0892719$, 29	$\mu = 0.00853659$, $\sigma = 0.0919984$, 100	DNF	DNF	DNF	DNF

Table G.6: Summary of the results from the PDE model (Metric=SIFT)

PDE results (SIFT)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
KNIGHT_STANDING	$\mu = 46.0599$, $\sigma = 36.2823$, 29	$\mu = 18.8402$, $\sigma = 6.9269$, 29	DNF	DNF	DNF	DNF	DNF
KNIGHT_KNEELING	$\mu = 24.8682$, $\sigma = 41.1049$, 29	$\mu = 10.1846$, $\sigma = 7.0582$, 29	DNF	DNF	DNF	DNF	DNF
BMNOISE	$\mu = 41.0030$, $\sigma = 25.9861$, 100	$\mu = 43.3470$, $\sigma = 30.2342$, 29	$\mu = 54.1163$, $\sigma = 29.2887$, 30	DNF	DNF	DNF	DNF
2DWOODBOX	$\mu = 27.9815$, $\sigma = 26.2352$, 29	$\mu = 31.2308$, $\sigma = 35.2842$, 98	$\mu = 51.5691$, $\sigma = 60.2166$, 30	$\mu = 35.8192$, $\sigma = 30.9142$, 100	DNF	$\mu = 72.6262$, $\sigma = 42.7289$, 30	DNF
KNIGHT_FIGHTING	$\mu = 31.7066$, $\sigma = 38.6682$, 30	$\mu = 5.4551$, $\sigma = 6.4981$, 101	DNF	DNF	DNF	DNF	DNF
2DTEAPOT	$\mu = 22.5229$, $\sigma = 21.4515$, 29	$\mu = 32.8437$, $\sigma = 25.8253$, 30	$\mu = 33.3480$, $\sigma = 23.1241$, 30	$\mu = 39.0586$, $\sigma = 24.5535$, 100	DNF	$\mu = 67.9527$, $\sigma = 40.3325$, 30	DNF

Table G.7: Summary of the results from the PDE model (Metric=Euclidean)

PDE results (Euclidean)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DTEAPOT	$\mu = 948.288$, $\sigma = 450.641$, 29	$\mu = 1364.55$, $\sigma = 258.926$, 29	$\mu = 1692.27$, $\sigma = 298.578$, 30	$\mu = 2070.36$, $\sigma = 382.061$, 30	DNF	$\mu = 2683$, $\sigma = 530.778$, 30	DNF
2DWOODBOX	$\mu = 1231.71$, $\sigma = 733.139$, 29	$\mu = 1659.51$, $\sigma = 684.129$, 30	$\mu = 2009.75$, $\sigma = 867.716$, 30	$\mu = 2316.29$, $\sigma = 962.049$, 30	DNF	$\mu = 2578.72$, $\sigma = 873.579$, 30	DNF
BMNOISE	$\mu = 2672.76$, $\sigma = 1688.62$, 99	$\mu = 4183.71$, $\sigma = 1163.4$, 100	$\mu = 4295.7$, $\sigma = 1099.35$, 30	DNF	DNF	DNF	DNF
KNIGHT_FIGHTING	$\mu = 640.471$, $\sigma = 371.965$, 30	$\mu = 777.687$, $\sigma = 287.333$, 30	DNF	DNF	DNF	DNF	DNF
KNIGHT_KNEELING	$\mu = 611.86$, $\sigma = 350.642$, 30	$\mu = 699.022$, $\sigma = 210.032$, 30	DNF	DNF	DNF	DNF	DNF
KNIGHT_STANDING	$\mu = 783.56$, $\sigma = 586.938$, 30	$\mu = 737.501$, $\sigma = 307.674$, 30	DNF	DNF	DNF	DNF	DNF
SINECOS	$\mu = 60.3854$, $\sigma = 57.0433$, 30	$\mu = 105.368$, $\sigma = 80.6578$, 30	$\mu = 131.457$, $\sigma = 91.349$, 30	$\mu = 242.312$, $\sigma = 124.261$, 30	DNF	DNF	DNF

G.3.2 Numeric

Table G.8: Summary of the results from the pde_num model (Metric=taxi)

pde_num results (taxi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DWOODBOX	DNF	$\mu = 102433$, $\sigma = 62459.6$, 10	DNF	DNF	DNF	DNF	DNF
SINECOS	$\mu = 115.283$, $\sigma = 141.163$, 1	$\mu = 230.117$, $\sigma = 162.468$, 2	$\mu = 306.528$, $\sigma = 157.789$, 1	$\mu = 465.314$, $\sigma = 219.674$, 5	DNF	DNF	DNF

Table G.9: Summary of the results from the pde_num model (Metric=pdiff)

pde_num results (pdiff)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DWOODBOX	DNF	$\mu = 958.167$, $\sigma = 354.357$, 13	DNF	DNF	DNF	DNF	DNF
SINECOS	$\mu = 0.365681$, $\sigma = 0.501116$, 1	$\mu = 0.754253$, $\sigma = 0.555183$, 3	$\mu = 1.08923$, $\sigma = 0.678827$, 1	$\mu = 1.34563$, $\sigma = 0.637976$, 1	DNF	DNF	DNF

Table G.10: Summary of the results from the pde_num model (Metric=mi)

pde_num results (mi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DWOODBOX	DNF	$\mu = 0.630534$, $\sigma = 0.132305$, 10	DNF	DNF	DNF	DNF	DNF
SINECOS	DNF	$\mu = 0.00283554$, $\sigma = 0.0531742$, 10	$\mu = 0.0035693$, $\sigma = 0.0596369$, 10	$\mu = 0.0035693$, $\sigma = 0.0596369$, 1	DNF	DNF	DNF

Table G.11: Summary of the results from the pde_num model (Metric=SIFT)

pde_num results (SIFT)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DWOODBOX	DNF	$\mu = 51.4820$, $\sigma = 52.2911$, 10	DNF	DNF	DNF	DNF	DNF

Table G.12: Summary of the results from the pde_num model (Metric=Euclidean)

pde_num results (Euclidean)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DWOODBOX	DNF	$\mu = 1870.64$, $\sigma = 905.185$, 10	DNF	DNF	DNF	DNF	DNF
SINECOS	$\mu = 63.3008$, $\sigma = 76.381$, 1	$\mu = 116.826$, $\sigma = 79.1879$, 2	$\mu = 138.326$, $\sigma = 70.1546$, 1	$\mu = 207.814$, $\sigma = 92.9772$, 5	DNF	DNF	DNF

Appendix H

Comparison of results

In this appendix we present a complete comparison of our results from the various different methods (Chapter 11). Only the results from the configuration with the lowest mean error are considered. In these tables a negative value indicates that the first named method had the lower error and shows by how much the two methods differed.

H.1 PDE analytic Vs. PDE numeric

Table H.1: Results from PDE analytic compared to PDE numeric, mi

results of PDE analytic compared to PDE numeric (mi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DWOODBOX	miss2	0.01	miss2	miss2	missa	miss2	missa
SINECOS	miss2	0.01	0.00	miss1	missa	missa	missa

Table H.2: Results from PDE analytic compared to PDE numeric, Euclidean

results of PDE analytic compared to PDE numeric (Euclidean)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DWOODBOX	miss2	-211.13	miss2	miss2	missa	miss2	missa
SINECOS	-2.92	-11.46	-6.87	34.50	missa	missa	missa

H.2 Linear Vs. PDE analytic

Table H.3: Results from Linear compared to PDE analytic, mi

results of Linear compared to PDE analytic (mi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DWOODBOX	-0.14	-0.05	-0.03	-0.02	miss2	-0.02	missa
BMNOISE	-0.48	-0.25	-0.19	missa	missa	missa	missa
KNIGHT_FIGHTING	-0.14	-0.07	missa	missa	missa	missa	missa
KNIGHT_KNEELING	-0.18	-0.07	missa	missa	missa	missa	missa
KNIGHT_STANDING	-0.20	-0.07	missa	missa	missa	missa	missa
SINECOS	miss1	-0.00	-0.00	miss2	miss2	missa	missa

Table H.4: Results from Linear compared to PDE analytic, Euclidean

results of Linear compared to PDE analytic (Euclidean)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DTEAPOT	-277.32	-183.73	-148.37	-145.88	miss2	-237.56	missa
2DWOODBOX	-432.52	-332.25	-333.40	-332.45	miss2	-677.44	missa
BMNOISE	-826.69	-1147.88	-958.35	missa	missa	missa	missa
KNIGHT_FIGHTING	-243.72	-198.47	missa	missa	missa	missa	missa
KNIGHT_KNEELING	-224.96	-178.18	missa	missa	missa	missa	missa
KNIGHT_STANDING	-350.98	-177.30	missa	missa	missa	missa	missa
SINECOS	-45.04	-51.00	-17.55	-70.40	miss2	missa	missa

H.3 Linear Vs. PDE numeric

Table H.5: Results from Linear compared to PDE numeric, taxi

results of Linear compared to PDE numeric (taxi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DWOODBOX	miss2	−32616.10	miss2	miss2	miss2	miss2	missa
SINECOS	−87.98	−124.79	−58.70	−96.31	miss2	missa	missa

Table H.6: Results from Linear compared to PDE numeric, pdiff

results of Linear compared to PDE numeric (pdiff)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DWOODBOX	miss2	−101.16	miss2	miss2	miss2	miss2	missa
SINECOS	−0.30	−0.51	−0.23	−0.35	miss2	missa	missa

Table H.7: Results from Linear compared to PDE numeric, mi

results of Linear compared to PDE numeric (mi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DWOODBOX	miss2	−0.04	miss2	miss2	miss2	miss2	missa
SINECOS	missa	0.00	0.00	0.00	miss2	missa	missa

Table H.8: Results from Linear compared to PDE numeric, Euclidean

results of Linear compared to PDE numeric (Euclidean)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2DWOODBOX	miss2	-543.38	miss2	miss2	miss2	miss2	missa
SINECOS	-47.95	-62.46	-24.42	-35.90	miss2	missa	missa

H.4 Linear Vs. NURBS

Table H.9: Results from Linear compared to NURBS, mi

results of Linear compared to NURBS (mi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2dwoodbox	−0.02	−0.02	−0.02	−0.01	−0.01	−0.06	missa
Bmnoise	−0.08	−0.05	−0.18	missa	missa	missa	missa
Brush	−0.02	−0.02	−0.02	−0.02	−0.02	−0.01	−0.03
Chessboard	−0.16	−0.01	0.00	0.00	−0.04	missa	missa
Circle_3	−0.02	−0.02	−0.01	−0.00	−0.00	0.00	0.00
Circle_4	−0.03	−0.02	−0.01	−0.00	−0.00	0.00	0.00
Circle_5	−0.03	−0.02	−0.01	−0.01	−0.00	0.00	0.00
Expt03	−0.02	−0.05	missa	missa	missa	missa	missa
Faces	−0.00	−0.12	missa	missa	missa	missa	missa
Idris_fig8	−0.04	−0.03	−0.03	−0.02	−0.02	−0.02	−0.01
Idris_straight	−0.03	−0.02	−0.02	−0.02	−0.02	−0.02	−0.02
Knight_fighting	−0.04	−0.12	missa	missa	missa	missa	missa
Knight_kneeling	−0.03	−0.08	missa	missa	missa	missa	missa
Knight_standing	−0.02	−0.07	missa	missa	missa	missa	missa
Sinecos	missa	−0.01	0.00	−0.00	−0.08	missa	missa
Straight_1	−0.03	−0.02	−0.02	−0.01	−0.01	−0.00	0.00
Straight_2	−0.03	−0.02	−0.02	−0.01	−0.01	−0.00	0.00
Straight_3	−0.03	−0.03	−0.02	−0.01	−0.01	−0.01	−0.00
Straight_4	−0.03	−0.02	−0.02	−0.02	−0.01	−0.01	−0.00
Straight_5	−0.03	−0.02	−0.02	−0.02	−0.01	−0.01	−0.00
Straight_6	−0.03	−0.02	−0.02	−0.01	−0.01	−0.01	−0.00
Straight_7	−0.03	−0.02	−0.02	−0.02	−0.01	−0.01	−0.00
Straight_8	−0.03	−0.03	−0.02	−0.02	−0.02	−0.01	−0.00
Woodbox	−0.05	−0.03	−0.03	−0.02	−0.02	−0.02	−0.01

Table H.10: Results from Linear compared to NURBS, SIFT

results of Linear compared to NURBS (SIFT)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
Chessboard	missa	missa	missa	0.00	missa	missa	missa
Circle_3	0.52	−3.62	−1.82	−0.07	0.29	−0.55	−3.65
Circle_4	0.53	−4.63	−1.74	1.76	0.77	1.35	3.13
Circle_5	0.82	−4.36	−6.22	0.05	−2.41	−3.37	−2.00
Faces	−18.85	miss2	missa	missa	missa	missa	missa
Idris_circle	−2.84	−1.76	−0.04	−4.23	−5.35	−5.30	−4.12
Idris_fig8	−2.71	−1.93	−1.17	−3.95	−3.23	−4.33	−5.05
Idris_straight	−1.11	−2.14	−0.77	−1.62	−1.64	−1.15	−1.52
Straight_1	−1.99	−1.98	0.46	2.32	−1.00	4.32	0.09
Straight_2	−3.00	−2.37	−1.94	3.08	−2.38	−8.29	3.16
Straight_3	−3.48	−2.74	−2.33	−4.63	−3.02	−8.08	1.10
Straight_4	−3.26	−2.85	−2.84	−1.66	−1.90	−1.81	5.63
Straight_5	−3.74	−3.69	−3.08	0.82	−5.59	−5.98	6.18
Straight_6	−4.02	−4.06	−0.84	−1.48	−4.53	−5.92	−2.56
Straight_7	−3.81	−3.11	−0.55	−2.77	−5.98	−5.69	−5.99
Straight_8	−2.72	−2.80	−2.02	−3.70	−9.10	−1.45	3.46
Woodbox	miss2	miss2	1.58	5.11	4.00	−1.80	miss1

Table H.11: Results from Linear compared to NURBS, Euclidean

results of Linear compared to NURBS (Euclidean)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2dteapot	−134.29	−170.56	−180.04	−126.63	−104.30	−736.50	missa
2dwoodbox	−213.09	−283.29	−270.19	−241.09	−162.04	−3060.15	missa
Bmnoise	−239.11	−216.17	−7568.25	missa	missa	missa	missa
Brush	−67.57	−60.63	−66.05	−61.92	−72.52	−60.75	−2786.45

Continued On Next Page. . .

Table H.11 – Continued

results							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
Chessboard	−340.38	−138.09	−1025.90	21.30	−439.10	missa	missa
Circle_3	−148.38	−202.72	−240.22	−257.56	−282.74	−225.46	−196.04
Circle_4	−143.55	−207.79	−238.85	−272.02	−286.65	−220.94	−189.12
Circle_5	−152.24	−215.86	−253.84	−247.39	−268.53	−224.14	−186.79
Expt03	−91.49	−1456.07	missa	missa	missa	missa	missa
Faces	−390.54	−2480.93	missa	missa	missa	missa	missa
Idris_circle	−62.11	−125.19	−171.83	−193.51	−221.39	−242.24	−283.46
Idris_fig8	−77.38	−130.63	−177.81	−243.65	−244.52	−288.70	−271.68
Idris_straight	−51.50	−44.85	−44.85	−59.12	−68.56	−98.34	−91.81
Knight_fighting	−49.85	−104.38	missa	missa	missa	missa	missa
Knight_kneeling	−42.08	−81.95	missa	missa	missa	missa	missa
Knight_standing	−44.71	−77.16	missa	missa	missa	missa	missa
Sinecos	−9.99	−17.25	−5.84	11.38	−66.50	missa	missa
Straight_1	−69.23	−80.74	−89.71	−101.20	−122.88	−135.83	−144.80
Straight_2	−66.23	−75.12	−88.07	−98.61	−98.95	−107.51	−117.76
Straight_3	−60.84	−66.03	−72.69	−74.78	−79.60	−68.36	−59.72
Straight_4	−60.89	−64.38	−72.44	−83.39	−77.69	−71.53	−58.26
Straight_5	−60.77	−64.02	−74.01	−89.94	−84.55	−76.02	−65.21
Straight_6	−62.50	−65.30	−77.16	−82.99	−90.73	−76.28	−64.29
Straight_7	−60.65	−63.93	−74.96	−90.95	−86.86	−82.04	−64.44
Straight_8	−60.80	−63.83	−76.14	−85.25	−89.96	−98.08	−85.82
Woodbox	−113.32	−132.90	−140.35	−149.56	−151.21	−164.18	−190.21

H.5 NURBS Vs. PDE

Table H.12: Results from PDE analytic compared to NURBS, mi

results of PDE analytic compared to NURBS (mi)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2dteapot	0.12	0.05	0.01	0.00	miss1	−0.08	missa
2dwoodbox	0.12	0.03	0.01	0.01	miss1	−0.05	missa
Bmnoise	0.40	0.20	0.01	missa	missa	missa	missa
Knight_fighting	0.11	−0.05	missa	missa	missa	missa	missa
Knight_kneeling	0.15	−0.01	missa	missa	missa	missa	missa
Knight_standing	0.17	0.00	missa	missa	missa	missa	missa
Sinecos	miss2	−0.01	0.00	miss1	miss1	missa	missa

Table H.13: Results from PDE analytic compared to NURBS, Euclidean

results of PDE analytic compared to NURBS (Euclidean)							
Manifold	$g = 2$	$g = 5$	$g = 10$	$g = 20$	$g = 25$	$g = 50$	$g = 100$
2dteapot	143.03	13.17	−31.67	19.25	miss1	−498.94	missa
2dwoodbox	219.43	48.96	63.21	91.36	miss1	−2382.71	missa
Bmnoise	587.58	931.71	−6609.90	missa	missa	missa	missa
Knight_fighting	193.87	94.09	missa	missa	missa	missa	missa
Knight_kneeling	182.89	96.23	missa	missa	missa	missa	missa
Knight_standing	306.26	100.14	missa	missa	missa	missa	missa
Sinecos	35.05	33.76	11.71	81.78	miss1	missa	missa